DOCUMENT RESUME

ED 223 829 CE 034 359

AUTHOR	Jennings, Carol Ann
TITLE	BASIC Programming.
SPONS AGENCY	Arkansas State Dept. of Education, Little Rock. Div. of Vocational, Technical and Adult Education.
PUB DATE	Sep 82
NOTE	357p.; Not available in paper copy due to colored paper in original.
PUB TYPE	Guides - Classroom Use - Guides (For Teachers) (052)
EDRS PRICE	MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS	Behavioral Objectives; *Computers; *Computer Science Education; Curriculum Guides; *Data Processing; Instructional Materials; Learning Activities; Postsecondary Education; *Programing; *Programing Languages; Secondary Education; Technical Education
IDENTIFIERS	*BASIC Programing Language

ABSTRACT

Designed for use by both secondary- and postsecondary-level business teachers, this curriculum guide consists of 10 units of instructional materials dealing with Beginners All-Purpose Symbol Instruction Code (BASIC) programing. Topics of the individual lessons are numbering BASIC programs and using the PRINT, END, and REM statements; system commands; constants, variables, arithmetic operations, and the LET statement; methods of input, including READ, DATA, RESTORE, and INPUT statements; program control, including relational operators, GO TO statements, IF THEN statements, logical operators, and counters; program development cycle and flowcharting; formulated output--PRINT USING statement and TAB function; FOR/NEXT statements, built-in functions, and subroutines; and subscripted variables, the dimension statement, and single- and double-dimensional arrays. Included in each unit are some or all of the following: performance objectives, suggested activities for instructors and students, information sheets, visual aids, assignment sheets, answers to assignment sheets, tests, test answers, and references. (MN)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED223829

BASIC PROGRAMMING

by

Carol Ann Jernings

Little Rock Public Schools
Parkview High School
2501 Barrow Road
Little Rock, AR 72204
(501) 225-6440

Developed in cooperation with

Arkansas Vocational and Technical Education
A Division of the Department of Education
Capitol Mall West - Little Rock, AR 72201

Tommie Butler, State Supervisor
Business Education

A Vocational Exemplary Project
Funded Under the Provision of
P.L. 94-482, Subpart 3, Section 132
Don Harlan, Supervisor
September, 1982

CE 034 359

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

☑ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

- Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

"PERMISSION TO REPRODUCE THIS
MATERIAL HAS BEEN GRANTED BY

J D Nichols

TO THE EDUCATIONAL RESOURCES
INFORMATION CENTER (ERIC)."

TABLE OF CONTENTS

UNIT		PAGE
I	Numbering BASIC Programs and Using the PRINT, END, and REM statements	1 - 28
II	System Commands	1 - 23
III	Constants, Variables, Arithmetic Operations, and the LET Statement	1 - 31
IV	Methods of Input--READ, DATA, RESTORE, and INPUT Statements	1 - 36
V	Program Control--Relational Operators, GO TO Statement, IF THEN Statement, Logical Operators, and Counters	1 - 35
VI	Program Development Cycle and Flowcharting	1 - 40
VII	Formatted Output--PRINT USING Statement and TAB Function	1 - 39
VIII	FOR/NEXT Statements	1 - 35
IX	Built-in Functions and Subroutines	1 - 37
X	Subscripted Variables, the Dimension Statement, and Single- and Double-Dimensional Arrays	1 - 39

NOTE: The curriculum guide is undergoing a final field test. Necessary corrections have been made in Units I through V. Users should be aware that there could be additional errors in Units VI through X.

PREFACE

Because of the tremendous changes in technology regarding the electronic handling of information, it has become possible and feasible to expect high schools and postsecondary business education classrooms to contain computers with both word and data processing capabilities. With the development of minicomputers and microcomputers, the prices have become reasonably affordable for many educational institutions. It is our philosophy that to provide realistic training opportunities for our students, our classrooms should duplicate as nearly as possible the equipment that is found in business and industry.

The new technological era is an exciting and challenging time for business education teachers. We are now faced with two tremendous challenges; that of retraining ourselves in the knowledge and skills necessary to provide training programs in the new technology and secondly, to persuade local school administratives and school boards to put the acquisition of minicomputers or microcomputers and appropriate software packages as a top priority. We cannot provide for our students adequate training opportunities with outdated knowledge, skills, and equipment.

Many progressive and innovative business teachers are now involved in training programs, both formal and self taught, and have made marvelous progress in developing information processing programs. Many far-sighted and progressive administrators and school boards have assisted the teachers in providing the newest technology in equipment for the business education classrooms.

We salute you, both business teachers and administrators, for the hard work, time, and money expended in this program development. A very special appreciation is given to Mrs. Carol Ann Jennings for the many, many, long hours, frustrations, and hard work that she has given to the development of this guide. As we turned the pages during the review sessions, we marveled at her great accomplishment.

This guide was developed in response to requests from Arkansas business teachers for assistance in developing their data processing programs. Teachers, take this guide and use it with the best wishes from the staff in the Division of Vocational and Technical Education who provided the opportunity and assisted in various ways during its development.

Tommie Butler
State Supervisor
Business Education

ACKNOWLEDGEMENTS

Thirteen Arkansas teachers who were involved in teaching BASIC Programming in their local schools were invited to serve on the Review/Evaluation Committee for the BASIC PROGRAMMING Curriculum Guide. Appreciation is extended to the following individuals who gave their time and effort in reviewing and evaluating the guide:

Marilyn Bonham	Green Forest High School
Pat Green	Fayetteville - West Campus
Joan Harper	Van Buren High School
Mary Jo Hood	Bentonville High School
Sara Huenefeld	Camden High School
Wanda Inman	Mills High School
Janet Lawrence	Oak Grove High School
Stan McKee	Newport High School
Dean Rossa	Jonesboro Area High School
Melessie Sims	White Hall High School
Martha Whatley	Jacksonville High School
Fran Williams	Rogers High School
Brenda Wilson	McClellan High School

A special "thank you" is extended to Cathy Martin for her help in developing the guide. Cathy devoted many extra hours to the project in typing, editing, printing, proofreading, flowcharting, and anything else needing to be done for the project. The guide would not have been completed without her help on the project.

Appreciation is also extended to Evedna Duran for her assistance in proofreading the guide, to Powell Hazzard for his help in reviewing the guide and writing solutions to programs, and to Gloria Venable for help in proofreading sections of the guide.

USE OF THIS CURRICULUM GUIDE

Introduction

BASIC PROGRAMMING is a curriculum guide. It is not intended to take the place of the instructor or to be used as a self-paced book.

It is important to remember that the BASIC language is not standardized. It is, therefore, necessary for the instructor to review the units before they are presented. The instructor will need to make any necessary changes for the units to be used with the computers available in his/her classroom.

Instructional Units

BASIC PROGRAMMING includes 10 units. Each instructional unit includes some or all of the basic components of a unit of instruction: performance objectives, suggested activities for instructors and students, information sheets, visual aids, assignments sheets, answers to assignments sheets, tests, and answers to the test. Units are planned for more than one lesson or class period of instruction.

Careful study of each instructional unit by the instructor will help to determine:

A. The amount of material that can be covered in each class period
B. The skills which must be demonstrated
    1. Supplies needed
    2. Equipment needed
    3. Amount of practice needed
    4. Amount of class time needed for demonstrations
C. Supplementary materials, such as pamphlets or filmstrips that can be ordered
D. Resource people who must be contacted

Objectives

Each unit of instruction is based on performance objectives. These objectives state the goals of the course, thus providing a sense of direction and accomplishment for the student.

Performance objectives are stated in two forms: the unit objective, stating the subject matter to be covered in a unit of instruction; and specific objectives, stating the student performance necessary to reach the unit objective.

Since the objectives of the unit provide direction for the teaching-learning process, it is important for the instructor and students to have a common understanding of the intent of the objectives. A limited number of performance terms have been used in the objectives for this curriculum to assist in promoting the effectiveness of the communication among all individuals using the materials.

The following is a list of performance terms and their synonyms which may have been used in this material:

Name

Label
List in writing
List orally
Letter
Record
Repeat
Give

Identify

Select
Match
Point out
Pick out
Choose
Locate
Mark

Describe

Define
Discuss in writing
Discuss orally
Interpret
Tell how
Tell what
Explain

Order

Arrange
Sequence
List in order
Classify
Divide
Isolate
Sort

Distinguish

Discriminate

Construct

Flowchart
Draw
Make
Build
Design
Formulate
Reproduce
Transcribe
Reduce
Increase
Figure

Demonstrate

Show your work
Show procedure
Perform an experiment
Perform the steps
Operate
Remove
Replace
Turn off/on
(Dis)assemble
(Dis)connect

Additional Terms Used

Code
RUN
Prepare
Make
Analyze
Calculate
Estimate
Plan
Observe
Compare
Tell
Teach

Enter
Evaluate
Complete
Determine
Solve
Write
Converse
Lead
State
Read
Perform

Reading of the objectives by the student should be followed by a class discussion to answer any questions concerning performance requirements for each instructional unit.

Instructors should feel free to add objectives which will fit the material to the needs of the students and community.

When instructors add objectives, they should remember to supply the needed information, assignment sheets, and criterion test.

Suggested Activities for the Instructor

Each unit of instruction has a suggested activities sheet outlining steps to follow in accomplishing specific objectives. Duties of instructors will vary according to the particular unit; however, for best use of the material they should include the following: provide the students with objective sheet, information sheet, assignment sheets; preview filmstrips, make transparencies, and arrange for resource materials and people; discuss unit and specific objectives and information sheets; give test. Instructors are encouraged to use any additional instructional activities and teaching methods to aid students in accomplishing the objectives. The references used in this guide are listed in unit one only.

Information Sheets

Information sheets provide content essential for meeting the cognitive (knowledge) objectives in the unit. The instructor will find that the information sheets serve as an excellent guide for presenting the background knowledge necessary to develop the skill specified in the unit objective.

Students should read the information sheets before the information is discussed in class. Students may take additional notes on the information sheets.

Transparency Masters

Transparency masters provide information in a special way. The students may see as well as hear the material being presented, thus reinforcing the learning process. Transparencies may present new information or they may reinforce information presented in the information sheets. They are particularly effective when identification is necessary.

Transparencies should be made and placed in the notebook where they will be immediately available for use. Transparencies direct the class's attention to the topic of discussion. They should be left on the screen only when topics shown are under discussion.

Assignment Sheets

Assignments sheets give direction to study and furnish practice for paper and pencil activities to develop the knowledges which are necessary prerequisities to skill development. These may be given to the student for completion in class or used as homework assignments. Answer sheets are provided which may be used by the student and/or instructor for checking student progress.

Test and Evaluation

Paper-pencil and performance tests have been constructed to measure student achievement of each objective listed in the unit of instruction. Individual test items may be pulled out and used as a short test to determine student achievement of a particular objective. This kind of testing may be used as a daily quiz and will help the instructor spot difficulties being encountered by students in their efforts to accomplish the unit objective. Test items for objectives added by the instructor should be constructed and added to the test.

Assignment Sheet Answers

Suggested solutions to the assignment sheets are provided with each unit. The solutions to the programs are only possible solutions. There are many other acceptable solutions to the programs assigned.

Test Answers

Test answers are provided for each unit. These may be used by the instructor and/or student for checking student achievement of the objectives.

BASIC PROGRAMMING

INSTRUCTIONAL/OCCUPATIONAL ANALYSIS

JOB TRAINING: What the Worker Should Be Able to Do (Psychomotor)	RELATED INFORMATION: What the Worker Should Know (Cognitive)
UNIT I: NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS	
	1. Terms and definitions
	2. Statement numbers
	3. PRINT statement
	4. END statement
	5. REM statement
6. Demonstrate the ability to code, enter, and RUN the PRINT statement using literals, commas, semicolons, blank spaces, and quotation marks	
7. Demonstrate the ability to use line numbers, the PRINT statement, the END statement, and the REM statement	
UNIT II: SYSTEM COMMANDS	
	1. Terms and definitions
	2. General information about system commands
	3. Common system commands
4. Demonstrate the use of the NEW, LIST, RUN, and SAVE command	
UNIT III: CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT	
	1. Terms and definitions
	2. Variables
	3. Constants
	4. Arithmetic operation

5. LET statement

6. Write algebraic expressions in BASIC notation

7. Solve BASIC expressions using the order of hierarchy and the LET statement

8. Label numeric variables, string variables, numeric constants, and string constants

9. Code, enter, and RUN programs to solve computations using the LET statements

UNIT IV: METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS

1. Terms and definitions

2. READ and DATA statements

3. RESTORE statement

4. INPUT statement

5. Write simple READ and DATA statements

6. Identify the values of variables in programs that include READ and DATA statements

7. Identify the value of variables in programs that include READ, DATA, and RESTORE statements

8. Code, enter, and RUN programs that use READ and DATA statements to store and use data within a program

9. Code, enter, and RUN a program using READ, DATA, and RESTORE statements

10. Code, enter, and RUN programs that produce descriptive messages for the input and require the values for the variables to be entered using INPUT statements

UNIT V: PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS

1. Terms and definitions
2. Relational operators
3. Unconditional branching--GO TO statement
4. Conditional branching--IF THEN statement
5. Logical operators
6. Counters
7. Write appropriate relational operators to correctly complete expressions
8. Write statements to transfer control using the GO TO statement and the IF THEN statement
9. Code, enter, and RUN programs using GO TO statements, IF THEN statements, logical operators, and counters

UNIT VI: PROGRAM DEVELOPMENT CYCLE AND FLOWCHARTING

1. Terms and definitions
2. Program development cycle
3. BASIC program flowcharting descriptions and their symbols
4. Structured flowcharting
5. Code, enter, and RUN programs using steps in flowcharts
6. Complete a program flowchart
7. Construct program flowcharts from coded programs
8. Demonstrate the ability to perform the steps in the program development cycle
9. Construct structured flowcharts in three basic designs

UNIT VII: FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION

1. Terms and definitions
2. PRINT USING statement
3. TAB function
4. Enter and RUN a program to demonstrate the output produced by numeric formatted fields
5. Enter and RUN a program to demonstrate the output produced by string formatted fields
6. Enter and RUN a program to demonstrate the output produced by three string formatted fields
7. Code, enter, and RUN programs using PRINT USING statements to produce formatted output for selected output
8. Flowchart, code, enter, and RUN programs using PRINT USING statements and TAB functions

UNIT VIII: FOR/NEXT STATEMENTS

1. Terms and definitions
2. FOR/NEXT statements
3. Nested FOR/NEXT statements
4. Flowcharting FOR/NEXT statements
5. Determine the results of programs using the FOR/NEXT statements
6. Code, enter, and RUN simple programs using FOR/NEXT statements
7. Flowchart, code, enter, and RUN programs using FOR/NEXT statements and nested FOR/NEXT statements

UNIT IX: BUILT-IN FUNCTIONS AND SUBROUTINES

1. Terms and definitions
2. Built-in functions
3. Subroutines
4. Write BASIC statements containing built-in functions
5. Write the results of built-in functions
6. Determine the output of programs containing subroutines
7. Flowchart, code, enter, and RUN programs using built-in functions and subroutines

UNIT X: SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS

1. Terms and definitions
2. Subscripted variables and single-dimensional arrays
3. DIM (dimension) statement
4. Processing data in an array
5. Double-dimensional arrays
6. Identify elements in single- and double-dimensional arrays by locations and values
7. Write the results of programs containing single- and double-dimensional arrays

8. Write statements to dimension, load, manipulate, and print single- and double-dimensional arrays

9. Flowchart, code, enter, and RUN programs containing single- and double-dimensional arrays

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

UNIT OBJECTIVE

After completion of this unit, the student should be able to select true statements about numbering programs, the PRINT statement, the END statement, and the REM statement. The student should also be able to demonstrate the ability to code, enter, and RUN a program using the PRINT, END, and REM statements. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with numbering BASIC programs and using the PRINT, END, and REM statements with their correct definitions.
2. Select true statements about numbering programs.
3. Select true statements about the PRINT statement.
4. Select true statements about the END statement.
5. Select true statements about the REM statement.
6. Demonstrate the ability to code, enter, and RUN the PRINT statement using literals, commas, semicolons, blank spaces, and quotation marks.
7. Demonstrate the ability to use line numbers, the PRINT statement, the END statement, and the REM statement.

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Provide student with precoded programs to be entered and RUN on a system.

VI. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

   A. Objective sheet

   B. Information sheet

   C. Transparency Master 1--REM Statements

   D. Assignment sheets

      1. Assignment Sheet #1--Demonstrate the ability to code, enter, and RUN the PRINT statement using literals, commas, semicolons, blank spaces, and quotation marks

      2. Assignment Sheet #2--Demonstrate the ability to use line numbers, the PRINT statement, the END statement, and the REM statement

   D. Answers to assignment sheets

   E. Test

   F. Answers to test

II. References:

   A. Albrecht, Bob, Don Inman, and Ramon Zamora. TRS-80 BASIC. New York: John Wiley and Sons, Inc., 1980.

B. Alonso, J.R.F. Simple BASIC Programs for Business Applications. New Jersey: Prentice-Hall, Inc., 1981.

C. Bitter, Gary G., and Wilson Y. Gateley. BASIC For Beginners. 2nd ed. New York: McGraw-Hill Book Company, 1978.

D. Bohl, Marilyn. Information Processing. Chicago: Science Research Associates, Inc., 1980.

E. Bosworth, Bruce, and Harry L. Nagel. Programming in BASIC for Business. Chicago: Science Research Associates, 1977.

F. Brown, Jerald R. Instant Freeze-Dried Computer Programming in BASIC. California: Dymax, 1977.

G. Computer Power, A First Course in Using the Computer. New York: McGraw-Hill Book Company, 1982.

H. Dock, Thomas V., and Edward Essick. Principles of Business Data Processing with BASIC. 3rd ed. Chicago: Science Research Associates, 1978.

I. Dwyer, Thomas A., and Margot Critchfield. A Bit of BASIC. Massachusetts: Addison-Wesley Publishing Company, 1980.

J. Gottfried, Byron S. Programming with BASIC. New York: McGraw-Hill Book Company, 1975.

K. Gustavson, Frances G., and Marian V. Sackson. Problem Solving and BASIC. Chicago: Science Research Associates, 1979.

L. Hirsch, Seymour C. BASIC Programming Self-Taught. Virginia: Reston Publishing Company, Inc, 1980.

M. Jennings, Carol Ann, and Sandy McDonald. Information Processing. Oklahoma: Mid-America Vocational Curriculum Consortium, Inc., 1981.

N. Katzan, Harry Jr. Introduction to Programming Languages. Pennsylvania: Auebach Publishers, Inc., 1973.

O. Koffman, Elliot B., and Frank L. Friedman. Problem Solving and Structured Programming in BASIC. Philippines: Addison-Wesley Publishing Company, Inc., 1979.

P. Ledin, George, Jr. A Structured Approach to Essential BASIC. California: Boyd and Fraser Publishing Company, 1980.

Q. Lien, David A. The BASIC Handbook. 2nd ed. California: Compusoft Publishing, 1981.

R. Logsdon, Tom. Programming In BASIC. California: Anaheim Publishing Company, 1978.

S. Lott, Richard W. BASIC with Business Applications. New York: John Wiley and Sons, Inc., 1977.

T. Mullish, Henry. A BASIC Approach to BASIC. New York: John Wiley and Sons Inc., 1976.

U. Murach, Mike. An Introduction to BASIC. Chicago: Science Research Associates, 1980.

V. Nelson, Dale E., Darrell V. Burras, Emma Jean Gillis, and Ron L. King. BASIC: A Simplified Structured Approach. Virginia: Reston Publishing Company Inc., 1981.

W. Orilia, Lawrence S. Introduction to Business Data Processing. New York: McGraw-Hill Book Company, 1979.

X. Parker, Alan J., and Val Silbey. BASIC for Business for the PDP-11. Virginia: Reston Publishing Company Inc., 1980.

Y. Peckham, Herbert D. BASIC, A Hands-on Method. New York: McGraw-Hill Book Company, 1978.

Z. Peluso, Anthony P., Charles R. Bauer, and Dalward J. Debruzzi. Basic BASIC Programming. Massachusetts: Addison-Wesley Publishing Company, 1972.

AA. Presley, Bruce, Stuart Hayes, Theodore Graham, and Harriet Morrill. A Guide to Programming in BASIC-PLUS. New Jersey: Lawrenceville School, 1980.

BB. Programming in BASIC. Massachusetts: Wang Laboratories, 1976.

CC. RSTS/E V7.0 Programming in BASIC-Plus. New Hampshire: Digital Equipment Corporation, 1979.

DD. Sack, John, and Judith Meadows Gabriel. Entering BASIC. 2nd. ed. Chicago: Science Research Associates, 1980.

EE. Sawatzky, Jasper J., and Shu-Jen Chen. Programming in BASIC-Plus. New York: John Wiley and Sons, Inc. 1981.

FF. Shelly, Gary B., and Thomas J. Cashman. Introduction to BASIC Programming. California: Anaheim Publishing Company, 1982.

GG. Silver, Gerald A., and Joan B. Silver. Simplified BASIC Programming. New York: McGraw-Hill Book Company, 1974.

HH. The BASIC Primer, Core Material. Massachusetts: Digital Equipment Corporation, 1979.

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

INFORMATION SHEET

I. Terms and definitions

A. BASIC - (Beginners All-Purpose Symbolic Instruction Code) a computer programming language that is useful for scientific and business problem solving often used with microcomputers and minicomputer systems

B. Program - a series of instructions that directs a computer in performing a specific task

C. Statement number - identifying numbers which precede BASIC statements in a program

D. Statement - a single BASIC instruction

E. RUN - the system command that directs the computer to execute the source program

(NOTE: Execute - to interpret a statement or series of statements and carry out the operations specified by the statements.)

(NOTE: Source program - the initial program solution written by the programmer.)

(NOTE: System command - instructions to the computer that tell the operating system what to do.)

F. PRINT statement - the BASIC statement that causes the computer to produce human readable output

G. Output - means by which information is reported from computer storage in a form that can be understood by human beings or in a form suitable for use as input for another computer

H. Computation - to determine mathematically an amount or number

I. Literals (strings) - expressions, labels, headings, and/or terms made up of alphabetic, numeric, and/or special characters (*, %, $, (), etc.) enclosed in quotation marks

INFORMATION SHEET

J. Print zones (fields or columns) - one of the areas on the terminal where an answer may be displayed

(NOTE: Most print zones consist of 15 spaces and 5 zones per line, but computers may vary.)

K. Quote - quotation marks either single (`´) or double ("")

(NOTE: Rules for using quotation marks may vary on different computers.)

L. END statement - statement in a BASIC program that indicates termination of the program

M. REM statement - the statement that allows the programmer to insert remarks or comments throughout the program without interrupting the operation of the computer

N. Documentation - recorded facts regarding a program

O. LIST - system command used to display program lines on a terminal

II. Statement numbers

A. Each line of a program must begin with a statement number

B. Statement numbers must be whole numbers

C. Statement numbers are necessary to tell the computer the sequence of the program statements

D. Statement numbers are usually numbered in multiples of ten so that additional statements may be added if necessary

E. The available range of line numbers is from 1 to 32676

(NOTE: On some computers you may begin at zero; and the range may be larger or smaller than 32676. Refer to your instruction manual.)

F. If lines are keyed out of numeric order when keying the program, the computer will run the program in numeric order

INFORMATION SHEET

Example:

```
40 PRINT 10 + 20 + 5
10 REM THIS PROGRAM WILL ADD THREE NUMBERS AND
20 REM PRINT THE ANSWER
30 PRINT "THE ANSWER IS";
99 END

LIST

10 REM THIS PROGRAM WILL ADD THREE NUMBERS AND
20 REM PRINT THE ANSWER
30 PRINT "THE ANSWER IS";
40 PRINT 10 + 20 + 5
99 END

RUN

THE ANSWER IS 35
```

G. If two statements in a program have the same line number, the first one typed will be replaced by the second

Example:

```
10 PRINT "ARKANSAS"
10 PRINT "LITTLE ROCK, ARKANSAS"
99 END

LIST

10 PRINT "LITTLE ROCK, ARKANSAS"
99 END

RUN

LITTLE ROCK, ARKANSAS
```

(NOTE: To correct an error after a return has been entered, retype the line number and the corrected line.)

H. To delete an entire line, type the line number of the line to be deleted followed by a return

INFORMATION SHEET

Example:

```
10 PRINT "ARKANSAS"
20 PRINT "ARKANSA"
30 PRINT "  IS A"
40 PRINT "NATURAL"
99 END
20

LIST

10 PRINT "ARKANSAS"
30 PRINT "  IS A"
40 PRINT "NATURAL"
99 END

RUN

ARKANSAS
  IS A
NATURAL
```

III. PRINT statement

A. General Information

1. Provides printed output in a BASIC program

2. Three general ways of using the PRINT statement:

   a. Labeling and providing headings

      Example:

```
10 PRINT "SALES","TAX","TOTAL"

RUN

SALES          TAX            TOTAL
```

   b. Carrying out single computations

      Example:

```
10 PRINT 10 + 12

RUN

 22
```

INFORMATION SHEET

c. Showing the end result if the LET statement is used

(Note: The LET statement is covered in Unit III.)

Example:

```
10 LET A = 3
20 PRINT A

RUN

 3
```

3. General format of the PRINT statement:

```
a.                 10 PRINT "SALES","TAX","TOTAL"
                    |    |   ___________|__________
     1) Line number      |              |
                         |              |
     2) The word PRINT|  |              |
                                        |
     3) Headings or labels______________|
        enclosed in quotation
        marks
```

```
b.                  10 PRINT 10 + 12
                     |    |   __|___
     1) Line number       |     |
                          |     |
     2) The word PRINT____|     |
                                |
     3) Computation(s)__________|
```

```
c.                       10 PRINT A, 15
                          |    |  __|__
     1) Line number_______|    |    |
                               |    |
     2) The word PRINT_________|    |
                                    |
     3) Values and/or_______________|
        numbers
```

4. Normally the PRINT statement will cause the output to be printed on a new line

5. The word PRINT or PRINT " " generates a blank line

INFORMATION SHEET

Example:

```
10 PRINT "THIS WILL"
20 PRINT
30 PRINT "DOUBLE SPACE"
40 PRINT ""
50 PRINT "THESE TWO LINES"
99 END

RUN

THIS WILL

DOUBLE SPACE

THESE TWO LINES
```

B. Literals (strings) in the PRINT statement

1. Both alphabetic and numeric literals can be printed by placing the items to be printed in quotes

2. Blanks are considered a character when enclosed in quotes

   Example:

```
10 PRINT "BLACK RIBBONS:  STOCK 2,356"

RUN

BLACK RIBBONS:  STOCK 2,356
```

3. A numeric may be printed without the use of quotes as long as it does not contain any special characters such as commas

   Example:

```
10 PRINT "BLACK RIBBONS:  STOCK";2356

RUN

BLACK RIBBONS:  STOCK 2356
```

4. On some computers, single quotes or double quotes are considered equivalent

   a. If double quotes are used at the beginning of the line, double quotes should appear at the end of the line

      (NOTE: End quotes are not always required.)

INFORMATION SHEET

Example:

Correct Usage

```
10 PRINT "BASIC IS EASY TO LEARN"
```

Incorrect Usage

```
10 PRINT "BASIC IS EASY TO LEARN'
```

b. If a single quote is used at the beginning of the line, a single quote must appear at the end of the line

Example:

Correct Usage

```
10 PRINT `BASIC IS EASY TO LEARN'
```

Incorrect Usage

```
10 PRINT `BASIC IS EASY TO LEARN"
```

(NOTE: When single and double quotes are not equivalent, double quotes must enclose strings. Check with your system manager.)

5. Two ways to include quotation marks in the printed output:

   a. The phrase to be printed with quotation marks is enclosed with single quotes with double quotes around the entire string

   Example:

```
10 PRINT "HE SAID, `HELLO'"

RUN

HE SAID, `HELLO'
```

   (NOTE: This method must be used when double and single quotes are not interchangeable.)

   b. The phrase to be printed with quotation marks is enclosed with double quotes with single quotes around the entire string

INFORMATION SHEET

Example:

```
10 PRINT `HE SAID, "HELLO"'
99 END

RUN

HE SAID, "HELLO"
```

C. Commas in the PRINT statement

1. In a PRINT statement, the comma sets the spacing at a print zone width of approximately 15 spaces

   (NOTE: Spacing will vary with equipment used: the TRS-80 has 4 zones of 16 spaces each; the DEC PDP 11/34 has 5 zones of 14 spaces each; the Commodore PET has 4 zones of 10 spaces each or 8 zones of 10 spaces each.)

2. The use of the comma permits up to 5 print zones (fields or columns) of approximately 15 spaces for output

   (NOTE: This will vary with equipment used.)

   EXAMPLE OF USING FIVE PRINT ZONES

```
10 PRINT "ZONE 1","ZONE 2","ZONE 3","ZONE 4","ZONE 5"
```

   (NOTE: The output has added number fields.)

```
RUN

              1              1              1              1
1234567890123451234567890123451234567890123451234567890123456....

ZONE 1         ZONE 2         ZONE 3         ZONE 4         ZONE 5
```

3. Positive values are printed one space from the left margin to allow for the assumed positive sign

   (NOTE: Positive signs are assumed but may be used if desired.)

INFORMATION SHEET

Example:

```
10 PRINT -1,2,-3,4,-5
99 END

RUN
```

```
-1               2              -3               4              -5

               1               1               1               1
1234567890123451234567890123451234567890123451234567890123456..
```

4. If more than five print zones are required, the terminal will advance to the next print line

Example:

```
10 PRINT 1,2,3,4,5,6,7
99 END

RUN
```

```
 1               2               3               4               5
 6               7

               1               1               1               1
1234567890123451234567890123451234567890123451234567890123456..
```

5. When using the comma to print in print zones, the terminal will automatically skip to the next print zone when more than 15 characters are required

Example:

```
10 PRINT "ABCDEFGHIJKLMNOPQ", "RSTUV"
99 END

RUN

ABCDEFGHIJKLMNOPQ             RSTUV

ZONE 1         ZONE 2         ZONE 3
```

6. If a comma is placed at the end of a PRINT statement, the computer will print the next PRINT statement on the same line if there is enough room

INFORMATION SHEET

Example:

```
10 PRINT 5,
20 PRINT 6

RUN

 5             6
```

D. Semicolons in the PRINT statement

1. The use of the semicolon allows data to be packed on a print line

    a. When using the semicolon between positive numerical data in a PRINT statement, two blank spaces will result (one space is for the positive sign)

    Example:

    ```
    10 PRINT 1;2;3;4;5
    99 END

    RUN

     1  2  3  4  5
    ```

    b. When printing negative numerical data, only one blank space is left

    Example:

    ```
    10 PRINT -1;-2;-3;-4;-5
    99 END

    RUN

    -1 -2 -3 -4 -5
    ```

    (NOTE: Spacing on systems may vary.)

2. When using a semicolon between literals, no space is provided. If blank spaces are desired, they must be included within the quotes

INFORMATION SHEET

Example:

```
10 PRINT "UNIVERSITY OF";" ARKANSAS"
99 END

RUN

UNIVERSITY OF ARKANSAS
```

3. If a semicolon is placed at the end of the PRINT statement, the computer will print the next PRINT statement on the same line if there is enough room

Example:

```
Correct output                  Incorrect output

10 PRINT "UNIVERSITY OF";       10 PRINT "UNIVERSITY OF";
20 PRINT " ARKANSAS"            20 PRINT "ARKANSAS"
99 END                          99 END

RUN                             RUN

UNIVERSITY OF ARKANSAS          UNIVERSITY OFARKANSAS
```

4. When using the semicolon between literal and numerical data, one space is provided before the numbers if the number is positive and no space is provided if the number is negative

(NOTE: This occurs if the numerical data follows the literal.)

Example:

```
10 PRINT "THE ANSWER IS"; 100-10
20 PRINT "THE ANSWER IS"; 10-100
30 PRINT "THE ANSWER IS "; 10-100
99 END

RUN

THE ANSWER IS 90
THE ANSWER IS-90
THE ANSWER IS -90
```

5. When a numeric expression (positive or negative) comes before a literal and is separated by a semicolon, a single space is left in the printed output

INFORMATION SHEET

Example:

```
10 PRINT "THE ANSWER IS"; 50-10;"POUNDS"
20 PRINT "THE ANSWER IS "; 40-50; "POUNDS"
99 END

RUN

THE ANSWER IS 40 POUNDS
THE ANSWER IS -10 POUNDS
```

IV. END statement

A. On most computers, the END statement must be the last statement in all BASIC programs and must have the highest statement number

B. In some forms of BASIC, the program will execute without the END statement, but it is good programming practice to include this statement in the program

C. The general format of the END statement is:

```
                                999 END
                                  |   |
1.  Line number___________________|   |
                                      |
2.  The word END______________________|
```

V. REM statement (Transparency 1)

A. The REMARK statement allows remarks or comments to be made throughout the program without affecting processing

B. The general format of the REM statement is:

```
                                 10 REM
                                  |   |
1.  Line number___________________|   |
                                      |
2.  The word REM______________________|
```

C. These remarks do not appear in the output but are necessary for proper documentation of the program

D. The REM statement will appear in the listing of a program but not in the RUN

INFORMATION SHEET

Example:

```
10 REM THIS PROGRAM WAS WRITTEN BY CATHY MARTIN
20 REM IT ILLUSTRATES PRINTING A NUMERIC VALUE
30 REM WITH A DOLLAR SIGN AND A COMMA
40 PRINT "$1,250.50"
99 END

RUN

$1,250.50
```

E. The following should be included in the documentation:

1. Name of the program
2. Name of the programmer
3. Brief explanation of what the program does
4. Description of the variables used in the program

   (NOTE: Variables are covered in Unit III.)

F. A blank space can be used to make the program easier to read

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

ASSIGNMENT SHEET #1--Demonstrate the ability to code, enter, and RUN the PRINT statement using literals, commas, semicolons, blank spaces, and quotation marks

1. Directions: Using the spaces provided, code PRINT statements to print the following:

   a.

```
**************************
*                        *
*      YOUR NAME         *
*                        *
**************************
```

____

____

____

____

____

____

   b. SHE SAID: `I WILL GO'

____

____

   c. 24 -1 3,400 $700

____

____

   d. 24   -1   DON

   (NOTE: Use three separate PRINT statements.)

____

____

____

____

ASSIGNMENT SHEET #1

e. UNITED STATES OF AMERICA

(NOTE: Use two separate PRINT statements.)

____

____

____

____

f. BASIC

PROGRAMMING

____

____

____

g.

```
    I     I
  O I  X  I O
____I_____I____
    I     I
    I  X  I
____I_____I____
    I     I
    I     I X
    I     I
```

(NOTE: The form of the final output may vary.)

____

____

____

____

____

____

____

____

____

2. Enter and RUN the coded lines on a computer.

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

ASSIGNMENT SHEET #2--Demonstrate the ability to use line numbers, the PRINT statement, the END statement, and the REM statement

Directions: Code, enter, and RUN the program to print a student's name, address, telephone number, Social Security Number, and date of birth.

Use appropriate REM statements.

Separate the information in Column A and Column B with a comma.

Use your own information for the output.

Sample output:

```
     Column A                             Column B

               NAME:                   POWELL HAZZARD
            ADDRESS:                   2501 BARROW ROAD
      TELEPHONE NO.:                   (501) 225-6440
SOCIAL SECURITY NO.:                   430-99-9999
      DATE OF BIRTH:                   DECEMBER 24, 1962
```

(NOTE: The words "Column A" and "Column B" are not to be printed.)

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

1. a.

```
10 PRINT "************************"
20 PRINT "*                      *"
30 PRINT "*       YOUR NAME      *"
40 PRINT "*                      *"
50 PRINT "************************"
999 END
```

b.

```
10 PRINT "SHE SAID:  `I WILL GO'"
999 END
```

c.

```
10 PRINT 24;-1;"3,400";" $700"
999 END
```

d.

```
10 PRINT 24;
20 PRINT -1;
30 PRINT "DON"
99 END
```

e.

```
10 PRINT "UNITED STATES";
20 PRINT " OF AMERICA"
999 END
```

f.

```
10 PRINT "BASIC"
20 PRINT
30 PRINT "PROGRAMMING"
999 END
```

g.

```
10 PRINT"     I     I"
20 PRINT"  O  I  X  I  O"
30 PRINT"_____I_____I_____"
40 PRINT"     I     I"
50 PRINT"     I  X  I"
60 PRINT"_____I_____I_____"
70 PRINT"     I     I"
80 PRINT"     I     I  X"
90 PRINT"     I     I"
999 END
```

Assignment Sheet #2

```
10 REM THIS PROGRAM WAS WRITTEN BY POWELL HAZZARD
20 REM THIS PROGRAM PRINTS INFORMATION ABOUT POWELL HAZZARD
30 PRINT "                  NAME:","STUDENT NAME"
40 PRINT "               ADDRESS:","STUDENT ADDRESS"
50 PRINT "                       ","STUDENT CITY AND STATE"
60 PRINT "      TELEPHONE NUMBER:","STUDENT TELEPHONE NUMBER"
70 PRINT "SOCIAL SECURITY NUMBER:","STUDENT SOCIAL SECURITY NUMBER"
80 PRINT "         DATE OF BIRTH:","STUDENT DATE OF BIRTH"
99 END
```

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

NAME ______________________________

TEST

1. Match the terms on the right with the correct definitions:

_____a. Means by which information is reported from computer storage in a form that can be understood by human beings or in a form suitable for use as input for another computer

_____b. (Beginners All-Purpose Symbolic Instruction Code) a computer programming language that is useful for scientific and business problem solving often used with microcomputers and minicomputer systems

_____c. One of the areas on the terminal where an answer may be displayed

_____d. The BASIC statement that causes the computer to produce human readable output

_____e. The system command that directs the computer to execute the source program

_____f. Quotation marks either single (`') or double ("")

_____g. Identifying numbers which precede BASIC statements in a program

_____h. Statement in a BASIC program that indicates termination of the program

_____i. A single BASIC instruction

_____j. Used to determine mathematically an amount or number

_____k. Statement that allows the programmer to insert comments or remarks throughout the program without interrupting the operation of the computer

1. Statement number
2. PRINT statement
3. Output
4. Computation
5. Statement
6. PRINT zones
7. Quote
8. END statement
9. Literals
10. Program
11. RUN
12. REM statement
13. BASIC
14. Documentation
15. LIST

______l. Expressions, labels, headings, or terms made up of alphabetic, numeric, and/or special characters (*, $, %, etc.) enclosed within quotation marks

______m. A series of instructions that directs the computer in performing a specific task

______n. Recorded facts regarding a program

______o. System command used to display program lines on a terminal

2. Select true statements concerning numbering programs by placing an "X" in the appropriate blanks.

______a. If lines are placed out of numeric order when the program is keyed, the program will not run

______b. The available range of line numbers is usually 1 to 256

______c. Line numbers are used to identify statements but play no part in the order of their execution

______d. It is necessary to begin each line of a program with a statement number

______e. If two statements have the same line number, the first one typed will be replaced by the second one

3. Select true statements concerning the PRINT statement by placing an "X" in the appropriate blanks.

______a. The following will print FUTURE BUSINESS LEADERS OF AMERICA with correct spacing:

```
10 PRINT "FUTURE BUSINESS LEADERS";
20 PRINT "OF AMERICA"
99 END
```

______b. The REM statements do not appear in the RUN of the program

______c. The output from this program is the following:

```
10 PRINT "JOHN SMITH",
20 PRINT "28 WEST 8",
30 PRINT "LITTLE ROCK, AR 72204"
99 END

RUN

JOHN SMITH
28 WEST 8
LITTLE ROCK, AR 72204
```

______d. The following program will print:

```
THERE ARE 15 TREES
   PRINT
10∧"THERE ARE"; 20-5; "TREES"
```

______e. The following statements are valid BASIC statements:

```
10 PRINT "THIS IS A BASIC STATEMENT"
20 PRINT "THE STOCK NUMBER IS" $1,215
```

______f. If more than five PRINT zones are required, the printer will not print the values after the fifth zone

______g. Normally the PRINT statement will cause the output to be printed on a new line

______h. When a number is printed, there is one space allowed for the positive or negative sign

______i. If the computer is commanded to PRINT "5+5", the result will be 10

______j. The following program will print JOHN SAID, 'HELLO':

```
10 PRINT "JOHN SAID, 'HELLO'"
```

______k. To print a person's name, it must be enclosed in quotation marks

______l. The last program statement is usually the REM statement

4. Select true statements concerning the END statement by placing an "X" in the appropriate blanks.

______a. The END statement is not necessary on computers but is used for documentation

______b. On most computers, the END statement must have the highest line number in the program

______c. The general form of the END statement is END 999

5. Select true statements concerning the REM statement by placing an "X" in the appropriate blanks.

______a. The REM statement is necessary for the program to RUN ______b. The following program is used to produce the output:

```
THE ANSWER IS 30

10 REM "THE ANSWER IS",
20 PRINT 30
99 END
```

______c. REM statements do not appear in the listing of a program

______d. REM statements do not appear when the program is RUN

______e. Remarks are necessary for proper documentation of a program

______f. The following should be included in the documentation of the program:

1. Name of the program
2. Name of the programmer
3. Brief explanation of what the program does
4. Description of the variables used

6. Demonstrate the ability to code, enter, and RUN the PRINT statement using literals, commas, semicolons, blank spaces, and quotation marks.

7. Demonstrate the ability to use line numbers, the PRINT statement, the END statement, and the REM statement.

(NOTE: If activities 6 and 7 have not been accomplished prior to the test, ask your instructor when they should be completed.)

NUMBERING BASIC PROGRAMS AND USING THE PRINT, END, AND REM STATEMENTS
UNIT I

ANSWERS TO TEST

1. a. 3
   b. 13
   c. 6
   d. 2
   e. 11
   f. 7
   g. 1
   h. 8
   i. 5
   j. 4
   k. 12
   l. 9
   m. 10
   n. 14
   p. 15

2. d.
   e.

3. b.
   d.
   g.
   h.
   j.
   k.

4. b.

5. d.
   e.
   f.

6. Evaluated to the satisfaction of the instructor.

7. Evaluated to the satisfaction of the instructor.

SYSTEM COMMANDS
UNIT II

UNIT OBJECTIVE

After completion of this unit, the student should be able to list six system commands and select true statements concerning system commands. The student should also be able to demonstrate the use of the NEW, LIST, RUN, SAVE, DIR, OLD, DELETE, RENAME, REPLACE, and KILL (UNSAVE) commands when entering a BASIC program. This knowledge will be evidenced by correctly performing the procedures outlined on the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with system commands with the correct definitions.

2. List six system commands.

3. Select true statements concerning system commands.

4. Demonstrate the use of the NEW, LIST, RUN, SAVE, DIR, OLD, DELETE, RENAME, REPLACE, and KILL (UNSAVE) commands when entering a BASIC program.

SYSTEM COMMANDS
UNIT II

SUGGESTED ACTIVITIES

I. Review your own system manual before beginning this unit. Changes may need to be made to this unit to cover the system commands unique to your computer.

II. Provide student with objective sheet.

III. Provide student with information and assignment sheets.

IV. Discuss unit and specific objectives.

V. Discuss information and assignment sheets.

VI. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

- A. Objective sheet
- B. Information sheet
- C. Assignment sheet
    - 1. Assignment Sheet #1--Demonstrate the use of the NEW, LIST, RUN, SAVE, DIR, OLD, DELETE, RENAME, REPLACE, and KILL (UNSAVE) commands when entering a BASIC program
- D. Answers to assignment sheets
- E. Test
- F. Answers to test

II. References:

Refer to Unit I references.

SYSTEM COMMANDS
UNIT II

INFORMATION SHEET

(NOTE: The commands in this unit will vary from computer to computer. Refer to your system manual and make any necessary changes before beginning this unit.)

I. Terms and definitions

A. System Commands - instructions to the computer that tell the operating system what to do

B. NEW - command that clears out the workspace in a computer's memory and allows the programmer to name the workspace

(NOTE: On microcomputers, the command NEW erases the program in memory.)

C. Ready - prompt that tells the operator to proceed with the BASIC commands or statements

D. File - a group of related records stored together

E. OLD - command that places previously saved program into current working space

(NOTE: The command LOAD is used on microcomputers.)

F. DIRECTORY - command used to print the list of files the programmer has created and stored along with information about them

G. LIST - command used to display each program line in a file on a terminal

H. RUN - command which begins the execution of a BASIC program

I. Syntax error - an error in the grammar or in the form of a BASIC program

J. SAVE - command that causes an entire new program to be stored in line number order on disk

K. RENAME - command used to give a new name to an existing file

L. REPLACE - command used to store revised program statements in a previously stored program

INFORMATION SHEET

M. DELETE - command used to delete a specific statement or statements from a BASIC program

N. KILL (UNSAVE) - command used to delete a previously saved program from permanent storage

II. General information about system commands

A. System commands are instructions to the computer that tell the operating system what to do

B. System commands are never preceded by a line number

C. List of several common system commands:

1. NEW
2. OLD
3. SAVE
4. LIST
5. RUN
6. DIRECTORY
7. RENAME
8. REPLACE
9. DELETE
10. KILL (UNSAVE)

III. Common system commands

A. NEW

1. Up to six alphanumeric characters are allowed for the file name of a BASIC program on most computer systems

2. The program name should be descriptive of the purpose of the program

3. To create a new program, the operator types NEW and then the file name

4. When the computer prints READY, the programmer begins entering the program

(NOTE: A few computers accept NE or N as abbreviations for NEW.)

Example of creating a new file tax

```
Ready

NEW

NEW FILE NAME--TAX

Ready
```

INFORMATION SHEET

or

```
NEW TAX
```

(NOTE: The program is then entered.)

```
100 REM THIS PROGRAM CALCULATES TAX
110 REM
120 REM      AMOUNT = 25
130 REM      RATE = .04
140 REM      TAX = AMOUNT TIMES RATE
150 REM
170 REM
200 PRINT "THE TAX IS $";25 * .04
999 END
```

5. Some syntax errors are caught when the program is entered

Example:

```
10 PRNIT "THE TAX IS $";25 * .04
?Illegal verb at line 10

10 PRINT THE TAX IS $;25 * .04
?Syntax error at line 10
```

(NOTE: The line would be retyped correctly before entering the rest of the program.)

B. OLD

1. An OLD program is one stored in the computer memory at an earlier time

2. To call up an old program, type OLD and return; then type the name of the program

Example of calling up an old program:

```
Ready

OLD

OLD FILE NAME--PROB2

Ready
```

C. SAVE

1. SAVE command is used to store on disk the current program in the computer for later use

INFORMATION SHEET

2. To store a program in a permanent file, type the word SAVE after entering the program

   Example of saving a program:

```
NEW

NEW FILE NAME--TAX

Ready

100 REM THIS PROGRAM CALCULATES TAX
110 REM
120 REM      AMOUNT = 25
130 REM      RATE = .04
140 REM      TAX = AMOUNT TIMES RATE
150 REM
160 PRINT "THE TAX IS $";25 * .04
999 END

SAVE

Ready
```

D. LIST

1. The LIST command is used to display each program line in a file on a terminal

2. The statements are listed in numeric order

3. One or more lines may be listed at a time

   Example:

```
OLD

OLD FILE NAME--TAX

Ready

LIST

100 REM THIS PROGRAM CALCULATES TAX
110 REM
120 REM      AMOUNT = 25
130 REM      RATE = .04
140 REM      TAX = AMOUNT TIMES RATE
150 REM
160 PRINT "THE TAX IS $";25 * .04
999 END

Ready
```

INFORMATION SHEET

```
LIST 100-140

(NOTE:  List several lines at a time.)

100 REM THIS PROGRAM CALCULATES TAX
110 REM
120 REM      AMOUNT = 25
130 REM      RATE = .04
140 REM      TAX = AMOUNT TIMES RATE

Ready

LIST 160

(NOTE: List a single line at a time.)

Ready

160 PRINT "THE TAX IS $";25 * .04

Ready
```

E. RUN

1. System command to begin the execution of the BASIC program

Example:

```
OLD

OLD FILE NAME--TAX

Ready

LIST

100 REM THIS PROGRAM CALCULATES TAX
110 REM
120 REM      AMOUNT = 25
130 REM      RATE = .04
140 REM      TAX = AMOUNT TIMES RATE
150 REM
160 PRINT "THE TAX IS $";25 * .04
999 END

Ready
```

INFORMATION SHEET

```
RUN

THE TAX IS $ 1

Ready
```

or

```
RUN TAX

THE TAX IS $ 1

Ready
```

2. Syntax errors may be indicated during the RUN of the program

   Example:

```
NEW

NEW FILE NAME--SYTERR

Ready

10 REM THIS PROGRAM DEMONSTRATES THE SYNTAX
20 REM ERROR MESSAGE
30 PPNT "HELP"
99 END

RUN

?ILLEGAL STATEMENT AT LINE 30

Ready
```

3. In some computers, a line number may be included after the RUN command to specify a starting line other than the first one

   Example:

   RUN 100 - Would start execution at line 100

F. DIRECTORY

1. System command used to obtain a listing of saved programs on disks

2. Can be abbreviated as DIR

INFORMATION SHEET

Example:

```
DIR

 Name  .Ext     Size    Prot        Date    SY:[4,22]

TAX    .BAS      1     < 60>   1-Aug-82
```

G. RENAME

1. Gives a new name to the program the user is working on
2. Enables programmer to keep the old program stored under old name and also to have the same program under the new name
3. SAVE command entered after the RENAME command

Example:

```
OLD

OLD FILE NAME--TAX

Ready

RENAME TAX1

SAVE

Ready

DIR

 Name  .Ext     Size    Prot        Date    SY:[4,22]

TAX    .BAS      1     < 60>   1-Aug-82
TAX1   .BAS      1     < 60>   1-Aug-82
```

H. REPLACE

1. Saves a new program and erases a previously saved program with the same name
2. The new program takes the place of the old one

INFORMATION SHEET

Example:

```
NEW

NEW FILE NAME--PAY4

100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM     RATE PER HOUR = 4
130 REM     HOURS WORKED = 40
150 REM
160 REM    PAY = RATE PER HOUR X HOURS WORKED
170 REM
180 PRINT "GROSS PAY",4 * 40
999 END

SAVE

Ready

RUN

GROSS PAY      160

Ready

120 REM     RATE PER HOUR = 5
180 PRINT "GROSS PAY",5 * 40

REPLACE

Ready

LIST

100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM     RATE PER HOUR = 5
130 REM     HOURS WORKED = 40
140 REM
150 REM
160 REM     PAY = RATE PER HOUR X HOURS WORKED
170 REM
180 PRINT "GROSS PAY",5 * 40
999 END

RUN

GROSS PAY      200

Ready
```

INFORMATION SHEET

I. DELETE

1. Used to erase BASIC statements in a program
2. The general form is:
   a. DELETE line number-line number

Example:

```
100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM     RATE PER HOUR = 4
130 REM     HOURS WORKED = 40
140 REM
150 REM
160 REM  PAY = RATE PER HOUR X HOURS WORKED
170 REM
200 REM
210 REM
220 REM
230 PRINT "GROSS PAY",4 * 40
999 END

DELETE 210-220

Ready

LIST

100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM     RATE PER HOUR = 4
130 REM     HOURS WORKED = 40
140 REM
150 REM
160 REM  PAY = RATE PER HOUR X HOURS WORKED
170 REM
200 REM
230 PRINT "GROSS PAY",4 * 40
999 END
```

   b. DELETE line number

```
Ready

DELETE 200

Ready
```

INFORMATION SHEET

```
LIST

100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM      RATE PER HOUR = 4
130 REM      HOURS WORKED = 40
140 REM
150 REM
160 REM  PAY = RATE PER HOUR X HOURS WORKED
170 REM
230 PRINT "GROSS PAY",4 * 40
999 END

Ready
```

3. A blank statement having a line number can also be used to delete one line

Example:

```
Ready

LIST

100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM      RATE PER HOUR = 4
130 REM      HOURS WORKED = 40
140 REM
150 REM
160 REM    PAY = RATE PER HOUR X HOURS WORKED
170 REM
230 PRINT "GROSS PAY",4 * 40
999 END

Ready

170

LIST

100 REM THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM      RATE PER HOUR = 4
130 REM      HOURS WORKED = 40
140 REM
150 REM
160 REM    PAY = RATE PER HOUR X HOURS WORKED
230 PRINT "GROSS PAY",4 * 40
999 END

Ready
```

INFORMATION SHEET

J. KILL (UNSAVE)

1. The KILL (UNSAVE) command is used to delete a previously saved program from permanent storage

2. After the KILL (UNSAVE) command is used, the program can no longer be called up by the OLD command

Example:

```
DIR

 Name  .Ext     Size    Prot      Date      SY:[4,22]

TAX    .BAS      1     < 60>   1-Aug-82
TAX1   .BAS      1     < 60>   1-Aug-82

OLD

OLD FILE NAME--TAX

Ready

KILL (UNSAVE)

Ready

or

Ready

KILL (UNSAVE) TAX

Ready

DIR

 Name  .Ext     Size    Prot      Date      SY:[4,22]

TAX1   .BAS      1     < 60>   1-Aug-82
```

SYSTEM COMMANDS
UNIT II

ASSIGNMENT SHEET #1--Demonstrate the use of the NEW, LIST, RUN, and SAVE command

1. Enter the command to create a NEW program with the file name PROG1.

2. After the Ready prompt, enter the following program:

```
100 REM     THIS PROGRAM IS USED TO PRINT
110 REM     TEN COMMON SYSTEM COMMANDS
120 PRINT "    COMMON"
130 PRINT "SYSTEM COMMANDS"
140 PRINT
150 PRINT "NEW"
160 PRINT "LIST"
170 PRINT "DIRECTORY"
180 PRINT "RENAME"
190 PRINT "REPLACE"
200 PRINT "SAVE"
210 PRINT "OLD"
220 PRINT "RUN"
230 PRINT "DELETE"
240 PRINT "KILL (UNSAVE)"
999 END
```

3. Enter the command LIST.

4. Enter the command RUN.

5. Enter the command SAVE.

6. Obtain a DIRECTORY of your account or disk.

7. Call up PROG1 by use of the OLD command.

8. Delete line 120.

   LIST and RUN the program.

9. Use the REPLACE command to store your change.

10. Call up PROG1 and RENAME it PROG1A.

    (NOTE: Be sure to save after the program is renamed.)

11. Obtain a DIRECTORY of your account or disk.

    Two files should be listed in the directory.

ASSIGNMENT SHEET #1

12. Call up PROG1A and LIST line 130, reenter line 130 as "TEN COMMON SYSTEM COMMANDS", REPLACE the old line with the new one, and then LIST and RUN the program.

13. Delete the files PROG1 and PROG1A by use of the KILL (UNSAVE) command.

14. Obtain a DIRECTORY of your account or disk.

SYSTEM COMMANDS
UNIT II

SUGGESTED ANSWERS TO ASSIGNMENT SHEET

Assignment Sheet #1

1.
```
NEW
New file name--PROG1
```

2.
```
100 REM     THIS PROGRAM IS USED TO PRINT
110 REM     TEN COMMON SYSTEM COMMANDS
120 PRINT "    COMMON"
130 PRINT "SYSTEM COMMANDS"
140 PRINT
150 PRINT "NEW"
160 PRINT "LIST"
170 PRINT "DIRECTORY"
180 PRINT "RENAME"
190 PRINT "REPLACE"
200 PRINT "SAVE"
210 PRINT "OLD"
220 PRINT "RUN"
230 PRINT "DELETE"
240 PRINT "KILL (UNSAVE)"
999 END
```

3.
```
LIST
PROG1   10:57 AM      11-May-82
100 REM     THIS PROGRAM IS USED TO PRINT
110 REM     TEN COMMON SYSTEM COMMANDS
120 PRINT "    COMMON"
130 PRINT "SYSTEM COMMANDS"
140 PRINT
150 PRINT "NEW"
160 PRINT "LIST"
170 PRINT "DIRECTORY"
180 PRINT "RENAME"
190 PRINT "REPLACE"
200 PRINT "SAVE"
210 PRINT "OLD"
220 PRINT "RUN"
230 PRINT "DELETE"
240 PRINT "KILL (UNSAVE)"
999 END
```

4.
```
RUN
PROG1   10:59 AM       11-May-82
    COMMON
SYSTEM COMMANDS

NEW
LIST
DIRECTORY
RENAME
REPLACE
SAVE
OLD
RUN
DELETE
KILL (UNSAVE)

Ready
```

5.
```
SAVE
```

6.
```
DIR

Name  .Ext     Size    Prot        Date          SY:[4,22]

PROG1   .BAS  1        < 60>      11-May-82
```

7.
```
OLD
Old file name--PROG1

Ready
```

8.
```
DELETE 120

Ready

LIST
PROG1   11:15 AM       11-May-82
100 REM     THIS PROGRAM IS USED TO PRINT
110 REM     TEN COMMON SYSTEM COMMANDS
130 PRINT "SYSTEM COMMANDS"
140 PRINT
150 PRINT "NEW"
160 PRINT "LIST"
170 PRINT "DIRECTORY"
180 PRINT "RENAME"
190 PRINT "REPLACE"
200 PRINT "SAVE"
210 PRINT "OLD"
220 PRINT "RUN"
230 PRINT "DELETE"
240 PRINT "KILL (UNSAVE)"
999 END
```

```
     RUN
     PROG1   10:59 AM      11-May-82
     SYSTEM COMMANDS

     NEW
     LIST
     DIRECTORY
     RENAME
     REPLACE
     SAVE
     OLD
     RUN
     DELETE
     KILL (UNSAVE)

     Ready

 9.  REPLACE

10.  OLD
     Old file name--PROG1

     Ready

     RENAME PROG1A

     Ready

     SAVE

11.  Name    .Ext   Size    Prot       Date          SY:[4,22]
     PROG1   .BAS  1       < 60>      11-May-82
     PROG1A  .BAS  1       < 60>      11-May-82

12.  OLD
     Old file name--PROG1A

     Ready
     LIST 130
     PROG1A  11:23 AM            11-May-82
     130 PRINT "SYSTEM COMMANDS"

     Ready

     130 PRINT "TEN COMMON SYSTEM COMMANDS"
     REPLACE

     Ready
```

```
LIST

PROG1A   11:25 AM      11-May-82
100 REM     THIS PROGRAM IS USED TO PRINT
110 REM     TEN COMMON SYSTEM COMMANDS
130 PRINT "TEN COMMON SYSTEM COMMANDS"
140 PRINT
150 PRINT "NEW"
160 PRINT "LIST"
170 PRINT "DIRECTORY"
180 PRINT "RENAME"
190 PRINT "REPLACE"
200 PRINT "SAVE"
210 PRINT "OLD"
220 PRINT "RUN"
230 PRINT "DELETE"
240 PRINT "KILL (UNSAVE)"
999 END

RUN
PROG1A   11:26 AM      11-May-82
TEN COMMON SYSTEM COMMANDS

NEW
LIST
DIRECTORY
RENAME
REPLACE
SAVE
OLD
RUN
DELETE
KILL (UNSAVE)

Ready
```

13.

```
KILL (UNSAVE) PROG1

Ready

KILL (UNSAVE) PROG1A
```

14.

```
DIR

Name  .Ext     Size    Prot       Date           SY:[4,22]
```

SYSTEM COMMANDS
UNIT II

NAME_____________________________

TEST

1. Match the terms on the right with the correct definitions:

_____a. An error in the grammar or in the form of a BASIC program

_____b. Instructions to the computer that tell the operating system what to do

_____c. Command used to delete a specific statement or statements from a BASIC program

_____d. Command that clears out the workspace in a computer's memory and allows the programmer to name the workspace

_____e. Command used to print the statements in a file on a terminal in numeric order

_____f. Prompt that tells the operator to proceed with the BASIC commands or statements

_____g. Command used to give a new name to an existing file

_____h. A group of related records stored together

_____i. Command that causes an entire new program to be stored in line number order

_____j. Command that places previously saved programs into current working space

_____k. Command which begins the execution of a BASIC program

_____l. Command used to print the list of files the programmer has created and stored, along with information about them

_____m. Command used to delete a previously saved program from permanent storage

_____n. Command used to store revised program statements in a previously stored program

1. DELETE
2. DIRECTORY
3. File
4. LIST
5. NEW
6. OLD
7. Ready
8. RENAME
9. REPLACE
10. RUN
11. SAVE
12. Syntax error
13. System command
14. KILL (UNSAVE)

2. List 6 of the 10 system commands covered in this unit.

_____________________a.

_____________________b.

_____________________c.

_____________________d.

_____________________e.

_____________________f.

3. Select true statements concerning system commands by placing an "X" in the appropriate blanks.

_____a. System commands begin with line numbers

_____b. Up to 16 alphanumeric characters are allowed for a file name of a BASIC program on most computer systems

_____c. A new program can be created by entering the system command NEW followed by the name of the file

_____d. By typing OLD, the operator is able to call up a previously stored program

_____e. The DIRECTORY command is used to store a current program in the computer for later use

_____f. The REPLACE command is used to substitute a revised program for a previously stored program with the same name

_____g. To use the DIRECTORY command, type DIR on most systems

_____h. The LIST command is used to list in numeric order one or more lines of a program

_____i. The RUN command is used to execute a program

_____j. The DELETE command is only used to delete an entire program

_____k. After the KILL (UNSAVE) command is used, the program can be retrieved by the OLD command

4. Demonstrate the use of the NEW, LIST, RUN, SAVE, DIR, OLD, DELETE, RENAME, REPLACE, and KILL (UNSAVE) commands when entering a BASIC program.

   (NOTE: If this activity has not been accomplished prior to the test, ask your instructor when it should be completed.)

SYSTEM COMMANDS
UNIT II

ANSWERS TO TEST

1. a. 12
   b. 13
   c. 1
   d. 5
   e. 4
   f. 7
   g. 8
   h. 3
   i. 11
   j. 6
   k. 10
   l. 2
   m. 14
   n. 9

2. Any of the following are correct:

   NEW
   OLD
   SAVE
   LIST
   RUN
   DIRECTORY
   RENAME
   REPLACE
   DELETE
   KILL (UNSAVE)

3. c.
   d.
   f.
   g.
   h.
   i.

4. Evaluated to the satisfaction of the instructor.

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

UNIT OBJECTIVE

After completion of this unit, the student should be able to select valid variable names; select true statements about constants; list the symbols for arithmetic operations and the order of hierarchy; and select true statements about the LET statement. The student should also be able to write algebraic expressions in BASIC notation; solve BASIC expressions using the order of hierarchy and the LET statement; and code, enter, and RUN programs to solve computations using LET statements. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with constants, variables, arithmetic operations, and the LET statement.
2. Select valid variable names.
3. Select true statements about constants.
4. List the symbols for arithmetic operations and the order of hierarchy.
5. Select true statements about the LET statement.
6. Write algebraic expressions in BASIC notation.
7. Solve BASIC expressions using the order of hierarchy and the LET statement.
8. Label numeric variables, string variables, numeric constants, and string constants.
9. Code, enter, and RUN programs to solve computations using the LET statements.

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with additional review and exercises on algebraic expressions if needed.

III. Provide student with information and assignment sheets.

IV. Discuss unit and specific objectives.

V. Discuss information and assignment sheets.

VI. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:
    A. Objective sheet
    B. Information sheet
    C. Transparency Masters
        1. TM 1--Decimal Form and E Notation
        2. TM 2--Writing Algebraic Expressions in BASIC Form With Sequence of Operation
        3. TM 3--LET Statement
    C. Assignment sheets
        1. Assignment Sheet #1--Write algebraic expressions in BASIC notation
        2. Assignment Sheet #2--Solve BASIC expressions using the order of hierarchy and the LET statement
        3. Assignment Sheet #3--Label numeric variables, string variables, numeric constants, and string constants

4. Assignment Sheet #4--Code, enter, and RUN programs to solve computations using the LET statements

D. Answers to assignment sheets

E. Test

F. Answers to test

II. References:

Refer to Unit I references.

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

INFORMATION SHEET

I. Terms and definitions

A. Variables - items identified whose values can change in a program

B. Numeric data - numbers

C. Alphanumeric characters - letters and/or symbols includes alphabetic characters, numbers, and special characters such as $, %, ;, *, -, /, and blank characters

D. String variables - names used to address alphanumeric characters, letters, numbers, or special characters

E. Constants - numeric or string values that do not change within a program

F. Arithmetic operators - symbols used to perform mathematical computations

G. LET statement - a BASIC statement that begins with the word LET in which the value shown on the right-hand side of the equal sign is stored in the name given on the left-hand side

II. Variables

A. Variables are items whose values are changed in a program

B. The variable value is stored in the computer and located by a variable name

(NOTE: This variable name is actually the address of the storage location in which the variable is stored.)

C. Two kinds of variables

1. Numeric variables (containing numeric data)

2. String variables (containing alphanumeric data)

INFORMATION SHEET

D. Numeric variable names consist of one of the following two combinations:

1. One character - must be a letter
2. Two characters - the first must be a letter and the second must be a number (The second character cannot be a letter or a special character)

Examples of valid numeric variables: A, F2, Z5, B4, C0

Examples of invalid numeric variables: 6A, B33, B-, CA, A*

E. String variables

1. Any variable which is assigned an alphanumeric value (i.e., a variable containing letters, numbers, and/or special characters, as opposed to a numeric variable, which can contain only numbers)
2. String variable names consist of a letter or a single letter followed by a single digit, and a $ sign

Examples:

A$ B1$ C$

	Variable Name	Variable Value
Valid Variable Names	A	-5
	B7	7.32
	X$	"ACLM"
	B1$	"P1%"
Invalid Variable Names	A	"Jane"
	B	"$3.45"

(NOTE: Check your system manual for rules regarding variable names.)

F. It is useful to assign variable names that are associated with the variable

Example: T for Total, C for Counter, N$ for Name

III. Constants

A. A value that cannot change to another value within a program

INFORMATION SHEET

B. Two kinds of constants

1. Numeric constants

    a. No quotation marks required

    b. Leading plus and minus signs are acceptable

    c. Decimal points are acceptable

    d. Three forms

        1) Integer numbers (whole numbers)

        Examples:

        1 -200 0 2448

        2) Real numbers (decimal numbers)

        Examples:

        10.9 -20.34 0.01 2448.1

        3) E format (floating point, exponential, scientific format, scientific notation) (Transparency 1)

            a) Occurs when the result of arithmetic operations is greater than 6 digits

            b) Means to the 10th power (multiplied by 10)

            Examples:

            2000000 = 2.0E6 2.000000

            .00000234 = 2.34E-6 .000002.34

2. String constants

    a. Alphanumerical values that are not changed in the program

    b. Quotations marks are required

INFORMATION SHEET

Examples:

Valid constants: -5.321, 10, 2, "The answer is"

Invalid constants: $50, 2.3%, 6.5.7, 4,125, 19-, The answer is

(Note: The constants should be changed to: "$50", "2.3%", 6.57 or "6.5.7", "4,125" or 4125 "19-","The answer is".)

IV. Arithmetic operation (Transparency 2)

A. Arithmetic operators (symbols) must be used to direct the computer to perform a computation

B. The arithmetic operators (symbols) are:

1. Addition +
2. Subtraction -
3. Multiplication *
4. Division /
5. Raising to a power (exponentiation) **, ^, or ↑

C. Order of hierarchy:

1. ()

All operations within the parentheses are calculated first. If there are parentheses within parentheses, the computer works from the inside out

2. **, ^, ↑

All exponentiation calculations are performed next

3. * and /

Multiplication or division operations are then performed in the order in which they appear from left to right

4. + and -

Addition and subtraction operations are executed in the order in which they appear from left to right

INFORMATION SHEET

Example: Sequence of operations without parentheses

WHEN ADDITION, SUBTRACTION, MULTIPLICATION, AND DIVISION ARE INVOLVED WITHOUT PARENTHESES:

```
10 PRINT 3 + 8 / 2 * 4 - 5
99 END

RUN
 14
```

The order would be

$3 + \underline{8/2} * 4 - 5$

$3 + \underline{4 * 4} - 5$

$\underline{3 + 16} - 5$

$\underline{19 - 5}$

14

WHEN EXPONENTION, ADDITION, SUBTRACTION, AND DIVISION ARE INVOLVED WITHOUT PARENTHESES:

```
10 PRINT 4 / 2 ** 2 - 3 + 6 / 2
99 END

RUN

 1
```

The order would be

$4 / \underline{2 ** 2} - 3 + 6 / 2$

$\underline{4 / 4} - 3 + 6 / 2$

$1 - 3 + \underline{6 / 2}$

$\underline{1 - 3} + 3$

$\underline{-2 + 3}$

1

INFORMATION SHEET

WHEN ADDITION, SUBTRACTION, AND DIVISION ARE INVOLVED WITH PARENTHESES:

```
10 PRINT 2 *(400 - 200) / 2 + (500 / 2 + 6) / 2
99 END
```

```
RUN

 328
```

The order would be

$2 * \underline{(400 - 200)} / 2 + (500 / 2 + 6) / 2$

$2 * 200 / 2 + \underline{(500 / 2} + 6) / 2$

$2 * 200 / 2 + \underline{(250 + 6)} / 2$

$\underline{2 * 200} / 2 + 256 / 2$

$\underline{400 / 2} + 256 / 2$

$200 + \underline{256 / 2}$

$\underline{200 + 128}$

328

Examples of expressions with and without parentheses:

Without Parentheses	With Parentheses
9 * 8 + 10/2	9 * (8 + 10)/2
~~9 * 8 + 5~~ 72+10/2	9 * 18/2
72 + 5	~~9 * 9~~ 162/2
77	81

D. BASIC does not accept "implied" arithmetic operation (Transparency 2)

Example:

3(A+B) must be written as 3 * (A+B)

INFORMATION SHEET

Example of algebraic expressions written in BASIC notation:

Expression	BASIC
$B^2 - C + 3$	`B**2 - C + 3` or `B^2 - C + 3` or `B↑2 - C + 3`
$U + \frac{C^2}{D^2}$	`U + C^2/D^2` or `U + C↑2/D↑2` or `U + C**2/D**2`
$\frac{4}{2} - F X D$	`4/2 - F * D`

Examples with and without parentheses:

Expression	BASIC

$B^2 + X/2Y$	`B**2 + (X/(2 * Y))`
	`B^2 + (X/(2 * Y))`
	`B↑2 + (X/(2 * Y))`

$A + (1 + R)^2$	`A + (1 + R)^2` or `A + (1 + R)↑2` or `A + (1 + R)**2`
$(3 + A) X (B - 2)$	`(3 + A) * (B - 2)`
$\frac{K(L + M)^2}{4}$	`(K * (L + M)^2)/4` or `(K * (L + M)**2)/4` or `(K * (L + M)↑2)/4`

V. LET statement (Transparency 3)

A. The LET statement is sometimes referred to as the assignment statement

B. The LET statement evaluates the expression on the right side of the equal sign and assigns that value to the variable on the left side of the equal sign (= means replace not equal to in BASIC)

C. The LET statement assigns a variable (i.e., to place a specific value into a particular variable)

Example:

```
100 LET X = 25
```

INFORMATION SHEET

D. The statement is used to place the results of a formula into a particular variable

Example:

```
150 LET Y = 2 * 3/4
```

E. The right-hand side of the equals sign may contain variables, but these must have been assigned their proper values in previous statements in the program

Examples:

Correct use of the LET statement

```
10 LET A = 10
20 LET B = 2
30 LET C = 5
40 LET D = A * B + C
50 LET A = D
60 PRINT A
999 END

RUN

 25
```

(NOTE: The value of A keeps the last assigned value only.)

Incorrect use of the LET statement

```
10 LET X = Y + 3
20 LET Y = 1
```

(NOTE: Assign a value to a variable before it is used on the right-hand side of the equality sign.)

F. Most systems allow the program to drop the command LET

```
10 Y = 20
20 X = 2 + Y
```

INFORMATION SHEET

G. The LET statement is also used to assign characters (string) to a string variable name

Example:

```
LET B$ = "INVENTORY SUMMARY"

10 LET A$ = "THE SUM OF NUMBERS IS"
20 LET A = 1
30 LET B = 2
40 LET C = 3
50 LET A = A + B + C
60 PRINT A$;A

RUN

THE SUM OF THE NUMBERS IS 6
```

DECIMAL FORM	E NOTATION
1520000	1.52E + 06
152000	1.52E + 05
15200	1.52E + 04
1520	1.52E + 03
152	1.52E + 02
15.2	1.52E + 01
1.52	1.52
0.152	1.52E - 01
0.0152	1.52E - 02
0.00152	1.52E - 03
0.000152	1.52E - 04
0.0000152	1.52E - 05
0.00000152	1.52E - 06

TM 1

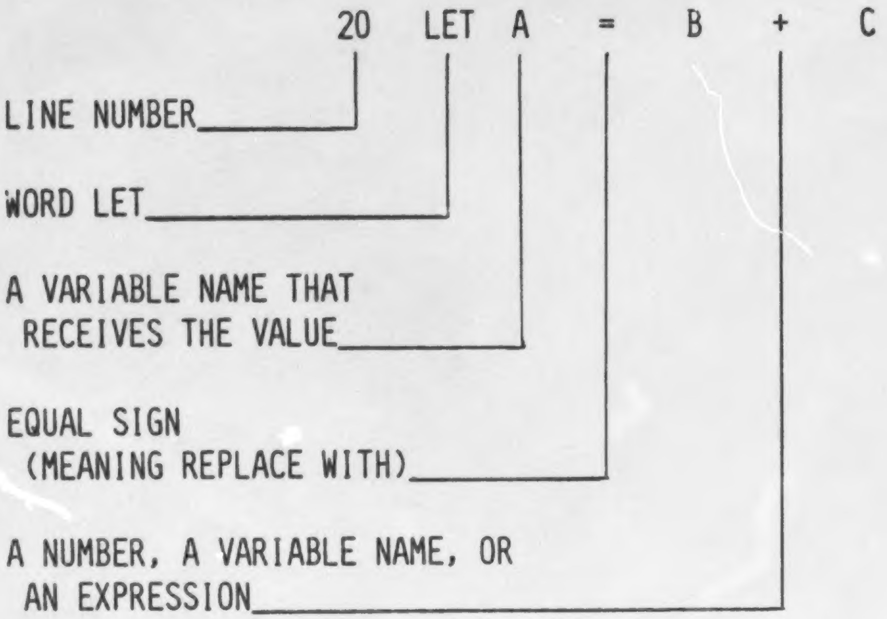
WRITING ALGEBRAIC EXPRESSIONS IN BASIC FORM WITH SEQUENCE OF OPERATION

EXPRESSION	BASIC	SEQUENCE
$B^2 - C + 3$	B**2 - C + 3 B ^ 2 - C + 3 B ↑ 2 - C + 3	$B \overset{1}{**} 2 \overset{2}{-} C \overset{3}{+} 3$
$U + \frac{C^2}{D^2}$	U + C**2/D**2	$U \overset{4}{+} C \overset{1}{**} 2 \overset{3}{/} D \overset{2}{**} 2$
4/2 - F X D	4/2 - F * D	$4 \overset{1}{/} 2 \overset{3}{-} F \overset{2}{*} D$

EXAMPLES WITH AND WITHOUT PARENTHESES:

EXPRESSION	BASIC	SEQUENCE
$B^2 + X/2Y$	B**2 + (X/(2 * Y))	$B \overset{3}{**} 2 \overset{4}{+} (X \overset{2}{/} (2 \overset{1}{*} Y))$
$A + (1 + R)^2$	A + (1 + R)^2 A + (1 + R)↑2 A + (1 + R)**2	$A \overset{3}{+} (1 \overset{1}{+} R) \overset{2}{**} 2$
(3 + A) * (B - 2)	(3 + A) * (B - 2)	$(3 \overset{1}{+} A) \overset{3}{*} (B \overset{2}{-} 2)$
$K \times (1 + M)^2/4$	K * (1 + M) ^ 2/4 K * (1 + M)**2/4 K * (1 + M) ↑ 2/4	$K \overset{3}{*} (1 \overset{1}{+} M) \overset{2}{**} 2 \overset{4}{/} 4$

LET STATEMENT


20 LET A = B + C
LINE NUMBER
WORD LET
A VARIABLE NAME THAT RECEIVES THE VALUE
EQUAL SIGN (MEANING REPLACE WITH)
A NUMBER, A VARIABLE NAME, OR AN EXPRESSION

```
10 LET A = 10
20 LET A = A + 1
30 LET A = A + 1
```

STATEMENT 10 ASSIGNS THE VALUE OF 10 TO THE VARIABLE A

STATEMENT 20 ASSIGNS THE VALUE OF 11 TO THE VARIABLE A (10 + 1)

STATEMENT 30 ASSIGNS A THE VALUE OF 12 (11 + 1)

THE VALUE OF 10 IS REPLACED BY 11 THEN THE VALUE OF 11 IS REPLACED BY 12

THE VARIABLE A ONLY RETAINS THE LAST VALUE

TM 3

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

ASSIGNMENT SHEET #1--Write algebraic expressions in BASIC notation

Directions: Write basic statements in the space provided for the following algebraic expressions.

a. ________________________________ $\frac{32}{4} + 6$

b. ________________________________ $\frac{\frac{1}{2} \text{ x } 10 + 5}{5}$

c. ________________________________ $\frac{1}{2} + \frac{36}{12}$

d. ________________________________ $C = \frac{X + 3}{\frac{Y - 4}{Z + 4}}$

e. ________________________________ $A = \frac{B}{\frac{C}{D}}$

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

ASSIGNMENT SHEET #2--Solve BASIC expressions using the order of hierarchy and the LET statement

1. What will be the output of the following statements:

Directions: Solve the following BASIC expression using the rules of order of hierarchy.

a. 100 PRINT 3 + 20 * 10 a. ______________

b. 120 PRINT 40 - 20/2 b. ______________

c. 130 PRINT 40 / 20*3 c. ______________

d. 140 PRINT 3**2+4/2 d. ______________

e. 150 PRINT 30 + 999/3 e. ______________

f. 999 END

2. Directions: Solve the following arithmetic operations, assign the following values to A, B, and C.

A = 18 B = 10 C = 2

a. D = ______________ a. D = (A + B + C)/2
b. D = ______________ b. D = A + B + C/2
c. D = ______________ c. D = A + B/C
d. D = ______________ d. D = (A+B)/C
e. D = ______________ e. D = A/B + C
f. D = ______________ f. D = A * (B + C)
g. D = ______________ g. D = A * B + C
h. D = ______________ h. D = A * (B * C)
i. D = ______________ i. D = (A + B) * C ** 2

3. Write the result of the following expression:

a. ((4 * 2) + 8) / 8

b. 4 * 2 + 8/8

c. A = 10 B = 30

A + B/2

d. A = 20 B = 40

(A + B)/2

ASSIGNMENT SHEET #2

4. Directions: Write the output of the following programs:

```
a. 10 LET A = 4
   20 LET B = 16
   30 LET C = 20
   40 LET D = C/A
   50 LET B = B/A
   60 LET C = A/B*C*2
   70 PRINT A;B;C;D
   99 END                              a. ________________
```

```
b. 10 LET A = 20 * 3
   20 PRINT 20;3;A
   99 END                              b. ________________
```

```
c. 10 PRINT 20;3;20*3
   99 END                              c. ________________
```

```
d. 10 LET A = 10
   20 LET B = 12
   30 LET C = A + B
   40 PRINT A;B;C
   99 END                              d. ________________
```

```
e. 10 LET A = 10
   20 LET B = 24
   30 LET C = A * B
   40 LET D = B/A
   50 PRINT A;B;C;D
   99 END                              e. ________________
```

```
f. 10 LET A = 10
   20 LET B = 24
   30 LET C = 20
   40 PRINT A*B/C
   99 END                              f. ________________
```

```
g. 10 PRINT 4 * 2 - 10 / 2
   99 END                              g. ________________
```

ASSIGNMENT SHEET #2

h.

```
10 LET A = 24
20 LET B = 6
30 PRINT A/(B + 6)
40 PRINT (A/B) + 6
50 END
```

h. ________________

5. Write the result of the following arithmetic operations in the space provided.

Y = 10   A = 8   B = 4

X = Y/A/B   a. ______
X = Y/(A/B) b. ______
X = (Y/A)/B c. ______

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

ASSIGNMENT SHEET #3--Label numeric variables, string variables, numeric constants, and string constants

Directions: Label the following parts of a BASIC program as numeric variables, string variables, numeric constants, or string constants by placing the appropriate letter in the blank provided:

NV - numeric variable
SV - string variable
NC - numeric constant
SC - string constant

Example:

```
10 LET C = 15
```

a. NV b. NC

```
10 LET A = 4
```

a. _____ b. _____

```
20 LET B = 5
```

c. _____ d. _____

```
30 LET N$ = "JOHN S. SMITH"
```

e. ______ f. ______

```
40 LET A$ = "2501 BARROW ROAD"
```

g. ______ h. ______

```
50 LET A1 = A*B
```

i. ______ j. ____ k. ____

```
60 PRINT N$;"LIVES AT","A$"
```

l. ______ m. ________ n. ____

```
70 PRINT N$;"IS"; A1; "YEARS OLD"
```

o. _____ p. ______ q. ___ r. _________

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

ASSIGNMENT SHEET #4--Code, enter, and RUN programs to solve computations using the LET statements

1. Directions: Write the following algebraic expressions in BASIC statements and then code, enter, and RUN a program to do the computations and print the results of each algebraic expression:

a. ____________ a. 100 + 2 =

b. ____________ b. 10 - 3 =

c. ____________ c. 425 x 10 =

d. ____________ d. 24 ÷ 3 =

e. ____________ e. 6^2 =

2. Directions: Code, enter, and RUN a program to multiply A times C and divide the amount by B. Assign the answer to the variable D:

Assign the following values

A = 60
B = 20
C = 10

3. Code, enter, and RUN a program to compute the following:

A = D + E
B = D + F
C = E + D + B

Assign values

D = 6 E = 4 F = 2.5

Print values of all variables

4. Code, enter, and RUN a program to average 4 grades; print the grades and average with appropriate headings.

GRADES

G1 -- 90
G2 -- 100
G3 -- 95
G4 -- 80

ASSIGNMENT SHEET #4

5. Code, enter, and RUN a program to compute the interest on a loan of 15,000 for 5 years at 12% interest rate. Use the REM, LET, and PRINT statements. The output should appear with appropriate headings.

   INTEREST = PRINCIPAL X RATE X TIME

6. Code, enter, and RUN a program to compute pay.

   The rate is 3.50 per hour. The hours are 40. The gross pay is equal to rate times the number of hours worked. Use your name as the name of the person to be paid.

7. Code, enter, and RUN a program to print the number of units in an ending inventory.

   Beginning units = 200
   Units received = 50
   Units released = 25

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

a. 32/4 + 6

b. (1/2 * 10 + 5)/5 or ((1/2) * 10 + 5)/5

c. 1/2 + 36/12 or (1/2) + 36/12

d. C = (X + 3)/((Y-4)/(Z + 4))

e. A = B/(C/D)

Assignment Sheet #2

1. a. 203
   b. 30
   c. 6
   d. 11
   e. 363
2. a. 15
   b. 29
   c. 23
   d. 14
   e. 3.8
   f. 216
   g. 182
   h. 360
   i. 112

3. a. 2

   b. 9

   c. 25

   d. 30

4. a. 4 4 40 5

   b. 20 3 60

   c. 20 3 60

   d. 10 12 22

   e. 10 24 240 2.4

   f. 12

   g. 3

   h. 2
      10

5. a. .3125

   b. 5

   c. .3125

Assignment Sheet #3

a. NV b. NC

c. NV d. NC

e. SV f. SC

g. SV h. SC

i. NV j. NV k. NV

l. SV m. SC n. SC

o. SV p. SC q. NV r. SC

Assignment Sheet #4

1. a. 100 + 2

   b. 10 - 3

   c. 425 * 10

   d. 24 / 3

   e. 6**2

```
10 PRINT 100 + 2
20 PRINT 10 - 3
30 PRINT 425 * 10
40 PRINT 24/3
50 PRINT 6**2
99 END

RUN
 102
 7
4250
 8
 36
```

2.

```
10 LET A=60
20 LET B=20
30 LET C=10
40 LET D=A*C/B
50 PRINT D
99 END

RUN
 120
```

3.

```
10 LET D=6
20 LET E=4
30 LET F=2.5
40 LET A=D+E
50 LET B=D+F
60 LET C=E+D+B
70 PRINT A;B;C;D;E
99 END

RUN
 10  8.5  18.5  6  4
```

```
4.  10 LET G1=90
    20 LET G2=100
    30 LET G3=95
    40 LET G4=80
    50 PRINT "G1=";G1
    60 PRINT "G2=";G2
    70 PRINT "G3=";G3
    80 PRINT "G4=";G4
    90 PRINT "AVERAGE = ";(G1+G2+G3+G4)/4
    99 END

    RUN
    G1= 90
    G2= 100
    G3= 95
    G4= 80
    AVERAGE = 91.25
```

```
5.  10 REM THIS PROGRAM WAS WRITTEN BY POWELL
    20 REM
    30 REM THIS PROGRAM COMPUTES THE INTEREST ON A LOAN
    40 REM
    50 REM                  LOAN     = 15000
    60 REM                  TIME     = 5 YEARS
    70 REM                  RATE     = 12%
    80 REM                  INTEREST = LOAN x RATE x TIME
    90 REM
    100 LET L=15000
    110 LET T=5
    120 LET R=.12
    130 LET I=L*R*T
    140 PRINT "LOAN","TIME","RATE","INTEREST"
    150 PRINT L,T,R,I
    999 END

    RUN
    LOAN           TIME           RATE           INTEREST
     15000          5              .12            9000
```

```
6.  10 LET P$="POWELL"
    20 LET R=3.50
    30 LET H=40
    40 LET G=H*R
    50 PRINT "NAME","RATE","HOURS","GROSS"
    60 PRINT P$,R,H,G
    99 END

    RUN
    NAME           RATE           HOURS          GROSS
    POWELL          3.5            40             140
```

7.

```
10 LET B = 200
20 LET U=50
30 LET S=25
40 LET E=(B+U)-S
50 PRINT "BEGINNING UNITS =";B
60 PRINT "UNITS RECEIVED =";U
70 PRINT "UNITS RELEASED =";S
80 PRINT "NEW UNITS TOTAL =";E
99 END

RUN
BEGINNING UNITS = 200
UNITS RECEIVED = 50
UNITS RELEASED = 25
NEW UNITS TOTAL = 225
```

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

NAME______________________________

TEST

1. Match the terms on the right with the correct definitions:

_____a. A numeric or string value that does not change within a program

_____b. Letters and/or symbols includes alphabetic characters, numbers, and special characters such as $, %, ;, *, -, /, and a blank character

_____c. A BASIC statement that begins with the word LET in which the value shown on the right-hand side on the equal sign is stored in the name given on the left-hand side

_____d. Items identified whose values change in a program

_____e. Symbols used to perform mathematical computations

_____f. Names used to address alphanumeric characters

_____g. Numbers

1. Alphanumeric character
2. Arithmetic operators
3. Constants
4. LET statement
5. Numeric data
6. String variables
7. Variable

2. Select valid variable names by placing an "X" in the appropriate blanks.

a. ____ B
b. ____ NAME
c. ____ I#
d. ____ A$

e. ____ C
f. ____ B*
g. ____ Z
h. ____ 3!

i. ____ B9
j. ____ AB14
k. ____ 14A
l. ____ 2

m. ____ AB
n. ____ R2
o. ____ (B)
p. ____ .B

q. ____ IB
r. ____ 1SR
s. ____ $R
t. ____ 3%

3. Select true statements about constants by placing an "X" in the appropriate blanks.

_____ a. The value of a constant changes throughout a program

_____ b. No quotation marks are needed for numeric constants

_____ c. No quotation marks are needed for string constants

_____ d. A numeric constant can be an integer, a real number, or an E format number

_____ e. $50 is a valid numeric constant

_____ f. -4.1 is a valid numeric constant

_____ g. "The Answer Is" is a valid string constant

_____ h. 4,125 is a valid numeric constant

_____ i. String constants may include letters, numbers, and/or special characters

_____ j. "JAN THOMAS" is a valid string constant

4. List the symbols for the following arithmetic operations and the order of hierarchy in the appropriate blanks.

Arithmetic Operation	Symbol
a. Addition	______
b. Subtraction	______
c. Multiplication	______
d. Division	______
e. Exponentiation	______

Order of hierarchy

f.

g.

h.

i.

5. Select true statements concerning the LET statement by placing an "X" in the appropriate blanks.

_____ a. The LET statement is also referred to as the assignment statement

_____ b. The word LET is required on all assignment statements

_____ c. The LET statement is only used for numeric data

_____ d. The LET statement evaluates the expression on the right-hand side and assigns that value to the variable on the left-hand side

_____ e. 10 LET 2 = Y is a valid LET statement

_____ f. 10 LET Y = 2 * 4/3 is a valid LET statement

_____ g. 10 LET N$ = "BILL JONES" is a valid LET statement

_____ h. A variable name can be used on the right-hand side of the equality as long as the variable has been previously assigned a value

_____ i. In the following program, the value printed for C is 10:

```
10 LET C = 10
20 LET A = 2
30 LET C = C * A
40 PRINT C
999 END
```

_____ j. The following is an example of the correct use of the LET statement:

```
10 LET A = 1
20 LET C = A + 3
```

6. Write algebraic expressions in BASIC notation.

7. Solve BASIC expressions using the order of hierarchy and the LET statement.

8. Label numeric variables, string variables, numeric constants, and string constants.

9. Code, enter, and RUN programs to solve computations using the LET statements.

(NOTE: If objectives 6 - 9 have not been accomplished prior to the test, ask your instructor when they should be completed.)

CONSTANTS, VARIABLES, ARITHMETIC OPERATIONS, AND THE LET STATEMENT
UNIT III

ANSWERS TO TEST

1. a. 3
   b. ~~6~~ 1
   c. 4
   d. 7
   e. 2
   f. ~~1~~ 6
   g. 5

2. a.
   d.
   e.
   g.
   i.
   n.

3. b.
   d.
   f.
   g.
   i.
   j.

4. a. +
   b. -
   c. *
   d. /
   e. **, ^, ↑
   f. Parentheses
   g. Exponentiation
   h. Multiplication and division (left to right)
   i. Addition and subtraction (left to right)

5. a.
   d.
   f.
   g.
   h.
   j.

6. Evaluated to the satisfaction of the instructor.

7. Evaluated to the satisfaction of the instructor.

8. Evaluated to the satisfaction of the instructor.

9. Evaluated to the satisfaction of the instructor.

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS

UNIT IV

UNIT OBJECTIVE

After completion of this unit, the student should be able to select true statements concerning the READ, DATA, and INPUT statements and distinguish between correct output and incorrect output in programs containing READ, DATA, and RESTORE statements. The student should also be able to identify the values of variables in programs that include READ, DATA, and RESTORE statements and code, enter, and RUN programs using READ, DATA, RESTORE, and INPUT statements. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with methods of input--READ, DATA, RESTORE, and INPUT statements--with their correct definitions.
2. Select true statements concerning the READ/DATA statement.
3. Distinguish between correct output and incorrect output in programs containing READ, DATA, and RESTORE statements.
4. Select true statements concerning the INPUT statements.
5. Write simple READ and DATA statements.
6. Identify the values of variables in programs that include READ and DATA statements.
7. Identify the value of variables in programs that include READ, DATA, and RESTORE statements.
8. Code, enter, and RUN programs using READ and DATA statements.
9. Code, enter, and RUN a program using READ, DATA, and RESTORE statements.
10. Code, enter, and RUN programs using the INPUT statement to enter data into BASIC programs.

METHODS OF INPUT--READ, DATA, RESTORE AND INPUT STATEMENTS
UNIT IV

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

  A. Objective sheet

  B. Information sheet

  C. Assignment sheets

    1. Assignment Sheet #1--Write simple READ and DATA statements

    2. Assignment Sheet #2--Identify the values of variables in programs that includes the READ and DATA statements

    3. Assignment Sheet #3--Identify the value of variables in programs that include READ, DATA, and RESTORE statements

    4. Assignment Sheet #4--Code, enter and RUN programs that use READ and DATA statements to store and use data within a program

    5. Assignment Sheet #5--Code, enter, and RUN a program using READ, DATA, and RESTORE statements

    6. Assignment Sheet #6--Code, enter, and RUN programs that produce descriptive messages for the input and require the values for the variables to be entered using INPUT statements

D. Answers to assignment sheets

E. Test

F. Answers to test

II. References:

Refer to Unit I references.

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

INFORMATION SHEET

I. Terms and definitions

A. READ and DATA statements - BASIC statements that cause data specified in DATA statements to be accessed (read) and assigned to specified variables

B. READ statement - BASIC statement that gives the variable name(s) for values stored in the program

C. DATA statement - BASIC statement that assigns the actual values to the variable(s) named in the READ statement

D. RESTORE statement - BASIC statement which enables the computer to use data in DATA statements more than once by resetting the data pointer back to the first piece of data in the first data line

E. Pointer - marker used to keep track of the next location of the data values

F. Data block - the collection of all values to be read by the READ statement into one sequential list

G. Nonexecutable statement - a statement that does not specify action but usually gives information

H. Sequential - one-after-the-other order

I. INPUT statement - BASIC statement that allows the operator to enter data values at the terminal during the execution (RUN) of the program

II. READ and DATA statements

A. General information

1. Method of entering data in a BASIC program

2. A program containing a READ statement must have at least one DATA statement

3. READ statements cause data specified in a DATA statement to be accessed (read) and assigned to specified variables

INFORMATION SHEET

a. READ statements give the variable names of the values stored in the program

```
30 READ A,B
```

b. DATA statements assign the actual values to the variable(s) named in the READ statement

```
30 DATA 10,2,"YES"
```

Example:

```
100 REM THIS PROGRAM READS IN AND PRINTS
110 REM OUT A NAME AND A NUMBER
200 READ A, N$
210 PRINT A, N$
220 DATA 14414, "CAROL JOHNSON"
999 END

RUN

 14414          CAROL JOHNSON
```

4. Any number of READ and DATA statements can appear in the program

5. Values may be read more than one at a time

6. Data stored by READ and DATA statements can be referenced at any point in the program by their assigned names (variables)

7. Data stored by READ and DATA statements can be manipulated, processed, and printed by reference to the variable names

8. More than one DATA statement may be used to satisfy one READ statement and more than one READ statement may be satisfied from one DATA statement

B. General format

1. READ statement

```
100 READ X, Y, Z$
```

a. Line number

b. The word READ

c. A list of variable names (separated by commas)

INFORMATION SHEET

2. DATA statement

```
200 DATA 10, -4, "THOMAS"
```

   a. Line number
   b. The word DATA
   c. A list of data values (separated by commas)

C. Rules concerning DATA values

1. Data values can be

   a. Numeric values

      1) Integer (3,2,-1)

      2) Decimal (2.4,-3.6)

      3) Floating point (exponential) (14.3E6)

   b. String values

      1) Quotation marks are generally optional

      2) Quotation marks are required when string values contain colons, commas, or blank spaces

      (NOTE: It is good practice to always enclose string values in quotation marks.)

```
100 READ A$, B$, C$
200 DATA "READ   ","DATA   ","RESTORE"
```

2. Numeric data cannot contain arithmetic expressions

D. Creation of data blocks

1. Computer collects all DATA statements in line number order to form the data block

2. As each READ is processed, the next available data value(s) is/are accessed from the data block

INFORMATION SHEET

Example:

```
DATA    {200 DATA 2,4,6,8,10
BLOCK   {210 DATA 3,5,7,9,11
```

Assembled data block

```
2,4,6,8,10,3,5,7,9,11
```

E. Order of execution of READ/DATA statement

1. Reads in order of statement number
2. Takes next available value(s) in data block and assigns to variable(s) in READ statement
3. Moves pointer to next item in data block and waits for another READ statement
4. Continues until last value in DATA statement has been processed
5. When READ statement tries to go beyond last value in DATA statement, the computer prints an OUT OF DATA message and stops

Example:

```
100 READ A,B,C
200 PRINT A;B;C
300 READ D,E,F
400 PRINT D;E;F
500 DATA 1,2,3
999 END
```

```
RUN

 1  2  3
?Out of data at line 300
```

F. Placement of READ/DATA statements

1. Can be placed anywhere in the program before the END statements because they are nonexecutable
2. Common to place just before the END statement

INFORMATION SHEET

Example:

```
10 REM DATA BEFORE READ   10 REM DATA IMMEDIATELY AFTER
20 DATA 6,7,8             20 REM READ
30 READ A,B,C             30 READ A,B,C
40 PRINT A;B;C            40 DATA 6,7,8
99 END                    50 PRINT A;B;C
                          99 END

RUN                       RUN
 6  7  8                   6  7  8
```

```
10 REM DATA JUST BEFORE END STATEMENT
20 READ A,B,C
30 PRINT A;B;C
40 DATA 6,7,8
99 END

RUN
 6  7  8
```

3. Computer matches variable(s) and data on a sequential one-for-one basis

   a. Breaking up READ statement does not change the output of the program

      Example:

```
10 READ A,B,C          10 READ A
20 DATA 1,2,3          20 READ B
30 PRINT A;B;C         30 READ C
99 END                 40 DATA 1,2,3
                       50 PRINT A;B;C
RUN                    99 END
 1  2  3
                       RUN
                        1  2  3
```

   b. Breaking up DATA statements does not change the output of the program

      Example:

```
10 READ A,B,C
20 DATA 1
30 DATA 2,3
40 PRINT A;B;C
50 END

RUN
 1  2  3
```

INFORMATION SHEET

4. Need to have at least as many values in the DATA statement as are variables in the READ statement

   a. If 3 variable names are given in a READ statement, there needs to be at least 3 values in DATA statements within the program or an error message will be printed

   b. If more values than variables are given in the DATA statement, the extra values are usually ignored

Excess Data

```
10 READ A,B,C
20 PRINT A;B;C
30 DATA 2,3,-4,5,3
99 END

RUN
 2  3 -4
```

Insufficient Data

```
10 READ A,B,C
20 PRINT A;B;C
30 DATA 2,3
99 END

RUN
?Out of data at line 10
```

(NOTE: Some computers print a message - EXTRA VALUES IGNORED.)

III. RESTORE statement

A. BASIC statement which enables computer to use data stored in DATA statement more than once

B. Resets the DATA pointer back to the first piece of data in the first DATA line

C. A subsequent READ statement will then read the data all over again

Example:

```
10 READ A,B
20 DATA 1,2
30 RESTORE
40 READ C,D
50 PRINT A,B,C,D
99 END

RUN
 1             2             1             2
```

INFORMATION SHEET

```
100 REM  THIS PROGRAM DEMONSTRATES THE USE OF THE
110 REM  RESTORE STATEMENT
120 READ A,B,C
130 PRINT "SUM ="; A+B+C
140 RESTORE
150 READ D,E,F
160 PRINT "PRODUCT ="; D*E*F
170 DATA 4,5,6
999 END

RUN

SUM = 15
PRODUCT = 120
```

IV. INPUT statement

A. General information

1. A BASIC command which requires data to be input during a program RUN (execution)

2. Allows operator to set a value for a variable during execution (RUN) of a program; it allows a "conversation" or "interaction" with the computer

   (NOTE: READ/DATA statements can be used in a program with the INPUT statement.)

```
100 REM  THE PROGRAM USES THE INPUT STATEMENT
110 REM  INSTEAD OF A READ/DATA STATEMENT
200 INPUT A
210 PRINT A
999 END

RUN
? 24
 24
```

3. There is no limit to the number of INPUT statements that can be placed in a program

INFORMATION SHEET

B. General Format

```
                              100 INPUT A, N$, B
                               |    |   ________
                               |    |      |
   1.  Line number_____________|    |      |
                                    |      |
   2.  The word INPUT_______________|      |
                                           |
   3.  Variable(s) name(s)_________________|
```

C. Advantages and disadvantages of INPUT statements

1. Advantages
    a. Allows user of the program to interact with the program and to try different data
    b. Is possible to RUN same program with different data values without having to change the program
2. Disadvantages
    a. Takes more time for the user of the program to RUN
    b. Not suited for large amounts of data
    c. Not as effective in terms of computer time as READ/DATA statements (execution of program is suspended until required data is entered)

D. Order of execution

1. Program is entered by programmer
2. RUN command is given by user
3. A ? will appear when values are needed for INPUT variables
4. The user will INPUT the needed values with commas between each
5. Execution stops until the needed value(s) is/are entered and a "return" or "enter" is pressed
    a. Error messages will appear indicating more data is needed

INFORMATION SHEET

Example:

```
10 INPUT A,B

RUN
? 1
? 2
```

(NOTE: On some computers, the data for the INPUT statement must be completely re-entered if a mistake is made.)

b. Program execution continues after correct format is typed

6. Program continues execution

E. Methods of using INPUT statement

1. Data alone can be input by operator through use of the INPUT statement

Example:

```
100 INPUT A

RUN
 ?
```

2. INPUT statements are often preceded by descriptive messages that tell the operator what information to input and in what order

(NOTE: Otherwise the "?" is useless.)

a. May be preceded with a PRINT statement identifying the data that should be entered

1) No punctuation mark before the variable causes a ? to appear on line under printed message

Example:

```
100 PRINT "ENTER AMOUNT OF PRINCIPAL ="
120 INPUT P
130 PRINT P
999 END

RUN

ENTER AMOUNT OF PRINCIPAL =
? 25000 (25000 entered by user)
 25000
```

INFORMATION SHEET

2) A semicolon after PRINT statements causes the ? from the INPUT statement to be on the same line

Example:

```
100 PRINT "ENTER AMOUNT OF PRINCIPAL =";
120 INPUT P
999 END

RUN

ENTER AMOUNT OF PRINCIPAL = ?
```

b. Many computers allow a description in the INPUT statement

```
100 REM  INTEREST PROBLEM USING INPUT STATEMENTS
110 INPUT "AMOUNT OF PRINCIPAL =";P
120 INPUT "INTEREST RATE =";R
130 LET I = R * P
140 PRINT "INTEREST =";I;"PRINCIPAL =";P; "RATE
=";R
999 END

RUN

AMOUNT OF PRINCIPAL =? 10000 (10000 entered by user)
INTEREST RATE =? .12 (.12 entered by user)
INTEREST = 1200 PRINCIPAL = 10000        RATE = .12
```

1) A semicolon before a variable name causes a question mark to be printed on the same line as the prompt

Example:

```
40 INPUT "WHAT IS YOUR FIRST NAME";N1$
```

The printout would appear as follows when N1$ is Sue

```
WHAT IS YOUR FIRST NAME? SUE
```

2) A comma before the variable name causes output on the same line but in the next print zone

INFORMATION SHEET

Example:

```
40 INPUT "WHAT IS YOUR FIRST NAME",N1$

RUN

WHAT IS YOUR FIRST NAME        ?SUE
```

(NOTE: This varies with computers.)

3) No punctuation before the variable causes the answer to be on the next line

Example:

```
40 INPUT "WHAT IS YOUR FIRST NAME" N1$

RUN
What is your FIRST NAME
? SUE
```

(NOTE: This varies with computers.)

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ASSIGNMENT SHEET #1--Write simple READ and DATA statements

Directions: In the space provided, write READ and corresponding DATA statements for the problems listed below.

In some problems, extra spaces are provided. Not all of the lines must be used.

a. Write a READ and DATA statement to assign 5 numeric values to 5 variables.

____

____

____

____

b. Write READ and DATA statements which assign 2 numeric values and 2 string values to 4 variables.

____

____

____

____

____

____

c. Write three separate READ and DATA statements with sufficient values for the three READ statements.

____

____

____

____

ASSIGNMENT SHEET #1

d. Write READ and DATA statements in which one READ statement is used to input data for 3 DATA statements.

____

____

____

____

e. Write 3 sets of READ and DATA statements. The first set will be used to read in one numeric value, the second set will be used to read in 2 string values, and the third set will be used to read in 3 numeric values and 2 string values.

____

____

____

____

____

____

f. Write READ and DATA statements that will assign the following:

A product name
A product number
Number of items in stock
Price of the product

____

____

____

____

____

____

____

____

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ASSIGNMENT SHEET #2--Identify the values of variables in programs that include READ and DATA statements

Directions: Write the output on the lines provided that results from the following programs containing READ and DATA statements:

a.

```
10 READ A,B,C
20 PRINT C;A;B
30 DATA 10,20,30,40,50,60
99 END
```

a. ______________________________

b.

```
10 READ A,B,C
20 DATA 10,100,12
30 PRINT A;B;C
99 END
```

b. ______________________________

c.

```
10 READ A,B,C
20 DATA 10,100
30 DATA 12
40 PRINT A;B;C
99 END
```

c. ______________________________

d.

```
10 DATA 10,100
20 READ A
30 DATA 12
40 READ B,C
50 PRINT A;B;C
99 END
```

d. ______________________________

e.

```
10 DATA 10,4,2
20 READ A,B,C
30 LET C = A+B+C
40 LET D = A + B + C/B
50 LET E = (A/B)*C+B
60 PRINT A;B;C;D;E
99 END
```

e. ______________________________

ASSIGNMENT SHEET #2

f.

```
10 READ A
20 PRINT A;
30 DATA -12
40 READ A
50 PRINT A
60 DATA 5
99 END
```

f. ____

g.

```
10 READ A,A
20 PRINT A;A;
30 PRINT A;
40 PRINT A
50 DATA 7,3
99 END
```

g. ____

h.

```
10 READ A1,B1,C1,D1
20 DATA 2,-6,7,4,24,0,12,6
30 READ F1,G1,H1,I1
40 DATA 6,3,14,59,84
50 PRINT A1;B1;C1;D1;F1;G1;H1;I1
99 END
```

h. ____

i.

```
10 DATA 2,4,6,8,10
20 READ T, T1
30 DATA 12,14,16
40 DATA 18
50 READ B,C,D
60 READ U,V,W,X
70 PRINT T;T1;B;C;D;U;V;W;X
99 END
```

i. ____

ASSIGNMENT SHEET #2

j.

```
10 READ A,B,C,D,E
20 DATA 2,4,6
30 DATA 1,3,5,7
40 READ F,G,H,I,J
50 DATA 8,10,12
60 DATA 9,11,13,15
70 READ K,L,M,N
80 PRINT A;B;C;D;E;F;G;H;I;J;K;L;M;N
99 END
```

j. ____

k.

```
10 READ N$,A,C,N1$,N2$,D
20 READ N2$,A1,N3$,A2,
30 READ N3$,A3,A4,A5,A6
40 DATA "RON",24,6
50 DATA "KEN","DON",6
60 DATA "ANN",8
70 DATA "ROBIN",4
80 DATA "MICHAEL",3
90 DATA "JANETTE",2
100 DATA 1,23
110 PRINT N$;A;C;N1$;N2$;D;N2$;A1;C1;N3$;A2;C2;N3$;A3;A4;A5
99 END
```

delete last comma

k. ____

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ASSIGNMENT SHEET #3--Identify the value of variables in programs that include READ, DATA, and RESTORE statements

Directions: Write the output of the following program containing READ, DATA, and RESTORE statements:

a.
```
10 READ A,B,C,D
20 RESTORE
30 READ A1,B1,C1,D1
40 DATA 1,2,3,4,5,6,7,8
50 PRINT A;B;C;D;
60 PRINT A1;B1;C1;D1;
99 END
```

a. ____

____

b.
```
10 READ A,A$,B,C,
20 RESTORE
30 READ A2,A$,A2,A3,A4
40 DATA 22,"MAY",14,15,1,"JUNE"
50 PRINT A;A$;B;C;A2;A$;A3;A4
99 END
```

b. ____

c.
```
10 READ A,B,A$,B$
20 RESTORE
30 READ C,D,C$,D$
40 RESTORE
50 READ E,F,E$,F$
60 DATA 2,6,"BAT"," BALL",3,4,"GLOVE"," BASE"
70 PRINT A;B;A$;B$
80 PRINT C;D;C$;D$
90 PRINT E;F;E$;F$
99 END
```

c. ____

____

d.
```
10 READ A,B,C
20 DATA 5,10,2
30 PRINT (A*B)/C,
40 RESTORE
50 READ B,A,C
60 PRINT (A*B)/C
70 RESTORE
80 READ C,A,B
```

ASSIGNMENT SHEET #3

```
90 PRINT (A*B)/C,
100 RESTORE
110 READ C,B,A
120 PRINT (A*B)/C
130 RESTORE
140 READ A,C,B
150 PRINT (A*B)/C,
160 DATA 6,4,2
999 END
```

d. ________________________________

e.

```
10 DATA 5,7,9
20 READ A,B,C
35 RESTORE
40 READ ~~A,B,C~~     B, A, C
45 PRINT A*A
99 END
```

e. ________________________________

f.

```
10 DATA 5,7,6,0
20 READ A,B,C
30 RESTORE
40 READ C,A,B
50 PRINT A*B
60 END
```

f. ________________________________

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ASSIGNMENT SHEET #4--Code, enter, and RUN programs that use READ and DATA statements to store and use data within a program

1. Using READ and DATA statements, code, enter, and RUN a program to compute and print the average of the following numbers:

   100 1000 2000 600 350

2. Using a READ and DATA statement, compute the conversion of Celsius temperature to Fahrenheit temperature

   Use the following formula:

   (Fahrenheit temperature = 1.8 * Celsius temperature + 32)

   The output should be similar to the following:

```
 CELSIUS          FAHRENHEIT

    0                 32
```

   Use the following celsius degrees as the data:
   10,0,-30,40

3. The Southern Computer Company pays its sales representatives a weekly salary of $300 plus 1 1/2% commission for all equipment sold during the pay period.

   Using READ and DATA statements, code, enter, and RUN a program to compute the commission earned and total weekly salary for each of the sales representatives.

   Print the sales representative's name, $300 salary, commission earned, and total weekly salary for each of the sales representatives.

Sales Representative	Sales
John Adams	20,000
Edward Brown	160,000
Paula Teed	4,500
Don Jones	56,000

ASSIGNMENT SHEET #4

4. A customer of the Arkansas Record Company buys 3 records priced at $7.15 each and 4 records priced at $9.00 each. Code, enter, and RUN a program to compute how much change the customer will receive if $100 is given to the clerk. Use a READ and a DATA statement.

5. Code, enter, and RUN a program to print a profit report for the XYZ Company.

   Read a value for the price per unit, cost per unit, and number of units sold.

   Print the number of units sold, the price per unit, and the cost per unit.

   Calculate total revenue and total cost. Calculate total profit by finding the difference in the total cost from the total revenue.

   Use the following data:

   Price $20.00
   Cost $16.00
   Number 4500

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ASSIGNMENT SHEET #5--Code, enter, and RUN a program using READ, DATA, and RESTORE statements

1. Code, enter, and RUN a program having single DATA and RESTORE statements for the following:

   Use the following DATA statement:

   200 DATA 2,4,6

   Assign the following numbers to the designated variables:

   Variable: A = 2, B = 4, C = 6, D = 2, E = 4, F = 6, G = 2,
   H = 4, I = 6

   Print the values of the variables A,B,C,D,E,F,G,H,I

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ASSIGNMENT SHEET #6--Code, enter, and RUN programs that produce descriptive messages for the input and require the values for the variables to be entered using INPUT statements

1. Code, enter, and RUN a program to enter 2 numeric values (A and B). Use the PRINT statement for the descriptive message. Compute the sum, difference, and product of the 2 numbers. The output should contain the values of A and B as well as a description of what result is being calculated (i.e., THE SUM OF 2 + 3 IS 5).

2. Code, enter, and RUN a program to enter the description, quantity, and unit price of an item being ordered. Use the INPUT statement for the descriptive messages. The operator should be prompted separately for each of the 3 items. There should be one INPUT for each variable.

   Calculate the total cost of the units ordered.

   Print the description, the quantity ordered, the unit price, and the total cost for the units ordered.

   Use the following data when RUNNING the program:

   DESCRIPTION - HAZELTINE 1552 TERMINALS
   QUANTITY - 5
   PRICE PER UNIT - $1,085

3. Modify program 2 to use a single INPUT statement to provide one descriptive message to obtain the values for all three variables.

4. Code, enter, and RUN a program to ask for a number in inches; then convert it to centimeters, and then to meters.

   Use the following formulas:

   1 inch = 2.54 centimeters
   100 centimeters = 1 meter

   The output should be similar to the following:

```
2 INCHES ARE EQUAL TO 5.08 CENTIMETERS
5.08 CENTIMETERS ARE EQUAL TO .0508 METERS
```

ASSIGNMENT SHEET #6

5. Code, enter, and RUN a program prompting for the beginning inventory and sales for each day of the week - Monday, Tuesday, Wednesday, Thursday, and Friday. The question mark should appear on the line under the descriptive message. The values for the variables representing the 5 days of the week should be entered through one INPUT statement.

   Compute the sales of the week and the ending inventory.

   Print the beginning inventory, the sales for each day, the total sales for the week, and the ending inventory.

6. Code, enter, and RUN a program which will calculate the amount of change a person has. Use INPUT statements to enter the number of pennies, nickels, dimes, and quarters. The descriptive message of the input should be contained in the INPUT statement.

   The output should be similar to the following:

   ```
   YOU HAVE $7.53 CHANGE
   ```

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

```
a.  10 READ A,B,C,D,E
    20 DATA 1,2,3,4,5

b.  10 READ A,B,C$,D$
    20 DATA 3,2,"CAT","DOG"

c.  10 READ A
    20 READ B
    30 READ C
    40 DATA 1,2,3

d.  10 READ A,B,C
    20 DATA 1
    30 DATA 2
    40 DATA 3

e.  10 READ A
    20 READ A$,B$
    30 READ B,C,D,N$,N1$
    40 DATA 10
    50 DATA "STUDENT","NUMBER"
    60 DATA 2,4,5,"GRADE","AVERAGE"

f.  10 READ P$,N,S,P
    20 DATA "DESKS",1024,24,296

    OR

    10 READ P$
    20 DATA "DESKS"
    30 READ N
    40 DATA 1024
    50 READ S
    60 DATA 24
    70 READ P
    80 DATA 296
```

Assignment Sheet #2

```
a.  30   10   20

b.  10   100   12

c.  10   100   12

d.  10   100   12
```

e. 10 4 16 18 44

f. -12 5

g. 3 3 3 3

h. 2 -6 7 4 24 0 12 6

i. 2 4 6 8 10 12 14 16 18

j. 2 4 6 1 3 5 7 8 10 12 9 11 13 15

k. %Data format error at line 20

Assignment Sheet #3

a. 1 2 3 4 1 2 3 4

b. 22 MAY 14 15 14 MAY 15 1

c.
```
2  6 BAT BALL
2  6 BAT BALL
2  6 BAT BALL
```

d.
```
25              25
 4               4
 1
```

e. 49

f. 42

Assignment Sheet #4

1.
```
10 READ A,B,C,D,E
20 LET T = A+B+C+D+E
30 LET A = T/5
40 PRINT "AVERAGE ="; A
50 DATA 100,1000,2000,600,300
99 END
```

2.
```
10 PRINT "CELSIUS","FAHRENHEIT"
20 READ A,B,C,D
30 LET F = 1.8 * A + 32
35 PRINT A,F
40 LET F = 1.8 * B + 32
45 PRINT B,F
50 LET F = 1.8 * C + 32
55 PRINT C,F
60 LET F = 1.8 * D + 32
65 PRINT D,F
70 DATA 10,0,-30,40
99 END
```

```
3.  100 REM  PROGRAM TO CALCULATE COMMISSION AND TOTAL SALARY
    110 REM  WEEKLY EARNING = $300 + 1 1/2% OF SALES
    120 REM  N1$, N2$, N3$, N4$ = EMPLOYEES NAME
    130 REM  S1, S2, S3, S4 = SALES
    140 REM
    150 PRINT "SOUTHERN COMPUTER COMPANY"
    160 PRINT
    170 PRINT "NAME","SALARY","COMMISSION","TOTAL WEEKLY SALARY"
    180 PRINT
    190 READ N1$,S1,N2$,S2,N3$,S3,N4$,S4
    200 LET S = 300
    210 PRINT N1$, S, S1 * .015, S + S1 * .015
    220 PRINT N2$, S, S2 * .015, S + S2 * .015
    230 PRINT N3$, S, S3 * .015, S + S3 * .015
    240 PRINT N4$, S, S4 * .015, S + S4 * .015
    250 DATA "JOHN ADAMS",20000,"EDWARD BROWN",160000
    260 DATA "PAULA TEED",4500,"DON JONES",56000
    999 END
```

```
4.  100 READ A,A1,B,B1,M
    110 LET C = M- (A*A1+B*B1)
    120 PRINT "CHANGE RECEIVED IS $";C
    140 DATA 3,7.15,4,9,100
    999 END
```

```
5.  100 REM PROGRAM COMPUTES TOTAL REVENUE, TOTAL COST, AND
    120 REM TOTAL PROFIT
    130 REM
    140 REM        SELLING PRICE PER UNIT = $20
    150 REM        COST PER UNIT = $7
    160 REM        UNITS SOLD = 400
    170 REM
    180 REM        P = PRICE PER UNIT
    190 REM        C = COST PER UNIT
    200 REM        U = NUMBER OF UNITS SOLD AND BOUGHT
    210 REM
    220 REM        T = P x U - C x U
    230 REM
    240 READ P,C,U
    250 PRINT
    260 PRINT "PROFIT REPORT FOR XYZ COMPANY"
    270 PRINT
    280 PRINT "NUMBER OF UNITS SOLD";U
    290 PRINT "PRICE PER UNIT";P,"COST PER UNIT";C
    300 PRINT "TOTAL REVENUE";P*U
    310 PRINT "LESS TOTAL COST";C*U
    320 PRINT
    330 PRINT
    340 PRINT "TOTAL PROFIT";P*U-C*U
    350 DATA 20,16,4500
    999 END
```

Assignment Sheet #5

1.
```
100 READ A,B,C
110 RESTORE
120 READ D,E,F
130 RESTORE
140 READ G,H,I
150 PRINT A;B;C;D;E;F;G;H;I
160 DATA 2,4,6
999 END
```

Assignment Sheet #6

1.
```
100 PRINT "INPUT THE VALUES OF A AND B"
110 INPUT A,B
120 S = A + B
130 D = A - B
140 P = A * B
150 PRINT "THE SUM OF";A;"AND";B;"IS";S
160 PRINT "THE DIFFERENCE IN";A;"AND";B;"IS ";D
170 PRINT "THE PRODUCT OF";A;"TIMES";B;"IS ";P
999 END
```

2.
```
100 INPUT "DESCRIPTION";D$
110 INPUT "QUANTITY";Q
120 INPUT "UNIT PRICE";P
130 C = Q * P
140 PRINT D$;Q;P;C
999 END
```

3.
```
10 INPUT "DESCRIPTION, QUANTITY, PRICE";D$,Q,P
20 C = Q * P
30 PRINT D$;Q;P;C
99 END
```

4.
```
100 INPUT "ENTER NUMBER IN INCHES";I
110 C = I * 2.54
120 M = C/100
200 PRINT I; "INCHES ARE EQUAL TO ";C;"CENTIMETERS"
210 PRINT C; "CENTIMETERS ARE EQUAL TO";M;"METERS"
999 END
```

5.
```
100 PRINT "WHAT IS BEGINNING INVENTORY"
120 INPUT I
130 PRINT "WHAT WERE SALES FOR EACH DAY OF THE WEEK"
140 INPUT M,T,W,T1,F
150 PRINT "BEGINNING INVENTORY";I;"UNITS"
160 PRINT
170 PRINT "                           DAILY SALES"
180 PRINT
190 PRINT "MON.","TUES.","WED.","THURS.","FRI."
200 PRINT
210 PRINT M,T,W,T1,F
```

```
     220 PRINT
     230 LET S = M + T + W + T1 + F
     240 PRINT "NUMBER OF UNITS SOLD THIS WEEK";S
     250 LET E = I - S
     260 PRINT "ENDING INVENTORY";E
     999 END
```

```
6.   110 PRINT "THIS PROGRAM CALCULATES THE TOTAL AMOUNT OF CHANGE"
     120 PRINT "A PERSON HAS.  AS THE QUESTIONS ARE ASKED, KEY IN THE
     130 PRINT "NUMBER OF EACH COIN TO BE ACCUMULATED"
     135 PRINT
     140 INPUT "NUMBER OF PENNIES",P
     150 INPUT "NUMBER OF NICKELS",N
     160 INPUT "NUMBER OF DIMES",D
     170 INPUT "NUMBER OF QUARTERS",Q
     180 C = P * .01 + N * .05 + D * .10 + Q * .25
     185 PRINT
     190 PRINT "YOUR TOTAL AMOUNT IS CHANGE IS $";C
     999 END
```

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

NAME ______________________________

TEST

1. Match the terms on the right with the correct definitions:

_____a. BASIC statement that gives the variable names of the values entered

_____b. Marker used to keep track of the next location of the data values

_____c. BASIC statements that cause data specified in DATA statements to be accessed (read) and assigned to specified variables

_____d. BASIC statement that allows the operator to enter data values at the terminal during execution (RUN) of the program

_____e. A statement that does not specify action but usually gives information

_____f. BASIC statement that assigns the actual values to the variable(s) named in the READ statement

_____g. One-after-the-other order

_____h. The collection of all of the values to be read into the computer into one sequential list

_____i. BASIC statement which enables the computer to use data in DATA statements more than once by resetting the data pointer back to the first piece of data in the first data line

1. Data block
2. DATA statement
3. INPUT statement
4. Nonexecutable statement
5. Pointer
6. READ and DATA statements
7. READ statement
8. RESTORE statement
9. Sequential

2. Select true statements concerning the READ and DATA statements by placing an "X" in the appropriate blanks.

_____a. The following is a valid numeric value in a DATA statement:

"TOM ANSON JOHNSON"

_____b. The data block for the following DATA statement would be 1,2,3,7,6,5,2,4

```
100 DATA 1,2,3
200 DATA 7,6
300 DATA 5
400 DATA 4,2
```

_____c. READ statements are used to give the variable names for the values stored in the program

_____d. DATA statements are used to actually assign the value to the variable in the READ statement

_____e. DATA statements are used without READ statements

_____f. When more values are listed in DATA statements than variables in READ statements, an OUT OF DATA MESSAGE is printed

_____g. Breaking up READ and DATA statements does not change the output of the program

_____h. When READ statements go beyond the last data value in the data block, the prompt assigns zeroes to the rest of the variables listed in the READ statement

_____i. DATA statements can be placed anywhere in a program except after the END statement

_____j. Only 8 DATA statements are allowed on most computers in one program

3. Distinguish between correct and incorrect output produced through READ, DATA, and RESTORE statements by placing an "X" to the left of the correct output.

Directions: For the four programs given below, correct and incorrect output have been listed. For each of these programs, place an "X" to the left of the correct output.

a.

```
10 READ A$
15 RESTORE
20 READ B$
25 RESTORE
30 READ C$
40 DATA "ANN ","ANNE ","ANNIE "
50 PRINT A$;B$;C$
99 END
```

_____1. ANN ANNE ANNIE

_____2. ANN ANN ANN

b.

```
10 READ A
20 READ B
25 RESTORE
30 READ C
40 READ D
50 PRINT A,B,C,D
60 DATA 1,2,3,4
99 END
```

_____1. 1 2 3 4

_____2. 1 2 1 2

c.

```
10 READ A
20 READ B
30 READ C
40 RESTORE
50 READ B
60 PRINT A;B;C
70 DATA 1,2,3
99 END
```

_____1. 1 2 3

_____2. 2 2 3

_____3. 1 1 3

d.

```
10 READ A$,A,G
20 READ B$,A1
30 PRINT A$;B$;A;A1
40 RESTORE
50 READ B$,A,G
60 PRINT B$,A,G
70 DATA "ELIZABETH",60,100,"ANN",61
80 DATA "JANICE",20,200,"LESLIE",4,100
99 END
```

```
_____1.  ELIZABETH      ANN          60          61
         ELIZABETH       60         100

_____2.  ELIZABETH      ANN          60          61
         JANICE          20         200

_____3.  ELIZABETH      ANN          60          100
         JANICE          20         200
         LESLIE           4         100
```

4. Select true statements concerning INPUT statements by placing an "X" in the appropriate blanks.

_____a. The use of INPUT statements allows program users to enter different data values without changing the original program

_____b. A ? indicates the computer has stopped the execution of the program and is awaiting needed data

_____c. Only 24 INPUT statements are allowed in each BASIC program

_____d. The INPUT statement is more efficient in terms of computer time than the READ/DATA statement _____e. No error messages are indicated if not enough data is entered or if the format is ~~correct~~ INCORRECT

_____f. A semicolon following the INPUT statement causes the ? to be printed on the line below the message

_____g. A PRINT statement can be used for a descriptive message for the INPUT statement

_____h. No punctuation before the variable name in an INPUT statement causes the ? to be printed on the same line as the descriptive message

_____i. If an error occurs when using the INPUT statement that stops the execution of the program, it will only continue execution again only after the RUN command is entered

_____j. INPUT statements often have descriptive messages for the user

5. Write simple READ and DATA statements.

6. Identify the values of variables in programs that include READ and DATA statements.

7. Identify the value of variables in programs that include READ, DATA, and RESTORE statements.

8. Code, enter, and RUN programs using READ and DATA statements.

9. Code, enter, and RUN a program using READ, DATA, and RESTORE statements.

10. Code, enter, and RUN programs using the INPUT statement to enter data into BASIC programs.

(NOTE: If objectives 5 - 10 have not been accomplished prior to the test, ask your instructor when they should be completed.)

METHODS OF INPUT--READ, DATA, RESTORE, AND INPUT STATEMENTS
UNIT IV

ANSWERS TO TEST

1. a. 7
   b. 5
   c. 6
   d. 3
   e. 4
   f. 2
   g. 9
   h. 1
   i. 8

2. c.
   d.
   g.
   i.

3. a. 2
   b. 2
   c. 3
   d. 1

4. a.
   b.
   g.
   j.

5. Evaluated to the satisfaction of the instructor.
6. Evaluated to the satisfaction of the instructor.
7. Evaluated to the satisfaction of the instructor.
8. Evaluated to the satisfaction of the instructor.
9. Evaluated to the satisfaction of the instructor.
10. Evaluated to the satisfaction of the instructor.

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS

UNIT V

UNIT OBJECTIVE

After completion of this unit, the student should be able to match symbols to numeric and string meanings, select true statements concerning the GO TO and the IF THEN statements, distinguish between true and false conditions containing logical operations, and list the steps in setting up a counter. The student should also be able to write appropriate relational operators to correctly complete expressions; write statements to transfer control using the GO TO and IF THEN statements; and code, enter, and RUN programs using the counter, GO TO, and IF THEN statements. This knowledge will be evidenced by successfully performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with relational operators, GO TO statements, IF THEN statements, logical operators, and counters.
2. Match symbols to numeric and string meanings.
3. Select true statements concerning the GO TO statement.
4. Select true statements concerning the IF THEN statement.
5. Distinguish between true and false conditions containing logical operators.
6. List the steps in setting up a counter.
7. Write appropriate relational operators to correctly complete expressions.
8. Write statements to transfer control using the GO TO and the IF THEN statements.
9. Code, enter, and RUN programs using the counter, GO TO, and IF THEN statements.

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:
   - A. Objective sheet
   - B. Information sheet
   - C. Transparency Master 1--Collating Sequence ASCII Code
   - D. Assignment sheets
     1. Assignment Sheet #1--Write appropriate relational operators to correctly complete expressions
     2. Assignment Sheet #2--Write statements to transfer control using the GO TO and the IF THEN statements
     3. Assignment Sheet #3--Code, enter, and RUN programs using the counter, GO TO, and IF THEN statements
   - D. Answers to assignment sheets
   - E. Test
   - F. Answers to test

II. References:

Refer to Unit I references.

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

INFORMATION SHEET

I. Terms and definitions

A. Relational operators - symbols used to make comparison of BASIC variables and expressions, and to test relationships in conditional statements, such as equal to (=)

B. Relational expression - a comparison of two numeric or string expressions that uses a relational operator that is evaluated as either TRUE or FALSE

C. Unconditional branching - statements that instruct the computer to branch to a specific statement number of an executable statement regardless of any condition

D. GO TO statement - BASIC statement that gives the line number of the statement to which an unconditional branch is made

E. Loop - sequence of instructions that is repeated until the terminating condition is reached

F. Looping - the process of repeating a series of statements in a program

G. Conditional branching - a transfer of program control that occurs only if specified conditions are met

H. IF THEN statement - conditional statement used to make certain transfers and to direct further processing based upon results of the test

I. Flag - a number or string placed at the end of a DATA statement to signal the end of data

J. Counters - variable locations that are used to count the number of times a certain event occurs (loop is completed)

K. Logical operator - used to make logical comparison with IF THEN statements for two or more relations

INFORMATION SHEET

L. Initializing - setting a variable to a specific starting value such as T = 0 or C = 1

M. Increment - to alter the value of a counter by a specified value

II. Relational operators

A. General information

1. Relational operators are symbols used to compare two numerical or two string expressions to form a relational expression

2. Relational expressions are either TRUE or FALSE

3. Expressions can be

   a. Variables

   b. Constants

   c. Combination of variables and constants (or expressions)

B. Six relational operators symbols:

Symbol	Numeric meaning	String meaning
=	equal to	equal
<	less than	precedes
>	greater than	follows
<= or =<	less than or equal to	precedes or is equal to
>= or =>	greater than or equal to	follows or equal to
<> or ><	not equal to	does not equal

INFORMATION SHEET

Example:

Relational Expression	Value
3 < 4	TRUE
4 < 3	FALSE
3 >= 3	TRUE
4 + 1 < 5 + 2	TRUE
4 + 1 > 5 + 2	FALSE
0 <> 1	TRUE

C. Use of relational operators

1. Meaning of relational numeric expressions

Operator	Example	Meaning
		Condition is TRUE if
=	A = B	A is equal to B
<	A < B	A is less than B
>	A > B	A is greater than B
<= or =<	A <= B	A is less than or equal to B
>= or =>	A >= B	A is greater than or equal to B
<> or ><	A <> B	A is not equal to B

2. Meaning of string relational expressions (Transparency 1)

Operator	Example	Meaning
=	A$ = B$	A$ and B$ are equal
<	A$ < B$	A$ precedes B$ in alphabetic order
>	A$ > B$	A$ follows B$ in alphabetic order
<=	A$ <= B$	A$ is equal to or precedes B$ in alphabetic order

INFORMATION SHEET

>=	A$ >= B$	A$ is equal to or follows B$ in alphabetic order
<>	A$ <> B$	A$ is not equal to B$

3. Comparison process of string relational expressions

   a. Compare from left to right a character at a time until a difference is found between the characters being compared

      Example:

      "ABC" < "AEF"

      1) Compares A and A

      2) Compares B and E

      3) Finds a difference in the two characters

      4) Condition is TRUE because B is less than E because B precedes E in alphabetic order

         "ABCDEFG" < "ABC" FALSE
         "ABCDEFG" > "ABC" TRUE

   b. Leading and trailing blanks are significant

      Example:

      " A" < "A"

   c. Upper and lower case letters are not equivalent

III. Unconditional branching--GO TO statement

   A. Alters sequential execution of a program

   B. Transfers program control to the line number specified

   C. Results in an unconditional branching (automatic)

INFORMATION SHEET

D. General format

1.

```
                                  100 GO TO 20
                                    |   ___|__  |
   Line number______________________|     |    |
                                          |    |
   Words GO TO____________________________|    |
     (or GOTO)                                 |
                                               |
   Line number control is transferred to_______|
```

2. Both the forms GO TO and GOTO are accepted on most computers

E. Directs control to any executable statement in program

1. Can be to a higher line number causing instructions between the two line numbers to be skipped

2. Can be to a lower line number causing instructions to be executed again

3. Can be transferred to END statement causing termination of the program

4. Can create an endless loop (infinite)

(NOTE: Check with instructor to know how to terminate execution of an endless loop; usually break key or "control" key and "C" key are pressed simultaneously.)

Example of endless loop:

```
10 LET A = 0
20 LET A = A + 1
30 PRINT A
40 GO TO 20
99 END

RUN

 1
 2
 3
^C

Ready
```

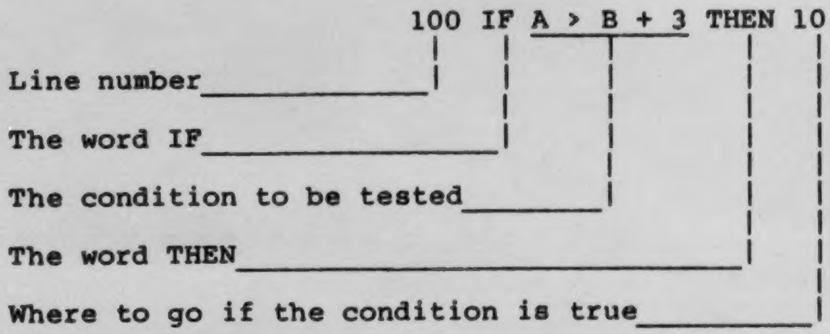
INFORMATION SHEET

IV. Conditional branching--IF THEN statement

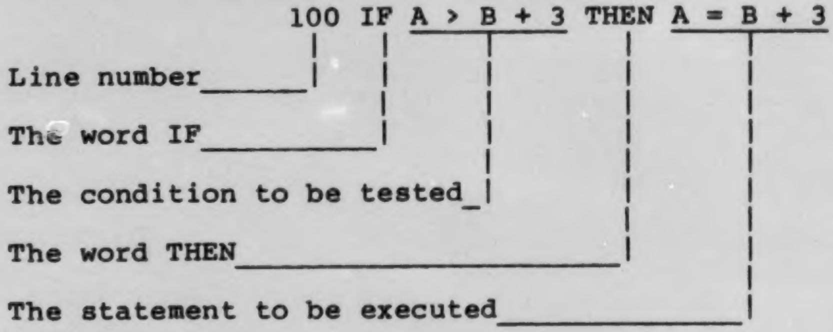
A. Directs the computer to examine a relational expression as TRUE or FALSE and transfers control to it if the relational expression is TRUE

B. General format

1.


100 IF A > B + 3 THEN 10
Line number
The word IF
The condition to be tested
The word THEN
Where to go if the condition is true

2.


100 IF A > B + 3 THEN A = B + 3
Line number
The word IF
The condition to be tested
The word THEN
The statement to be executed

3. Condition to be tested may be

a. Numbers (2,4,6)

b. Strings ("EXIT")

c. Variable names (A,B,C$)

d. BASIC expressions (A*B, (A+B/C))

C. Control passes

1. If condition is TRUE, the control passes to the command or statement number after THEN

2. If condition is FALSE, the computer ignores command after THEN and continues execution of the next line number

INFORMATION SHEET

Examples:

`IF A > B THEN C = D`	If A is greater than B, then make C = D
`IF A < B THEN N = B * C`	If A is less than B, then make N equal to B times C
`IF A <> B THEN 999`	If A is not equal to B, then go to statement 999

(NOTE: It is possible to write more than one statement on a line by following each statement with a colon.)

```
10 LET A = 5: T = 0: C = 0:
```

This is especially useful in IF THEN statements

```
100 IF A = B THEN PRINT "A = B": GO TO 999
```

D. Testing for end of data with IF THEN statement

1. To solve the problem of the OUT OF DATA message, a flag is placed at the end of the data line

2. IF A = -999 THEN 999

   Example:

```
10 REM  USE OF THE IF THEN STATEMENT TO TEST
20 REM  FOR VALID DATA
30 READ A
40 IF A = -999 THEN 999
50 PRINT A;
55 GOTO 30
60 DATA 1,2,3,20,-30,-999
999 END

RUN
 1  2  3  20 -30
```

V. Logical operators

A. Used to make logical comparisons with IF THEN statement to test for two or more relations

INFORMATION SHEET

B. Operators used

1. AND

2. OR

   (NOTE: Other operators are used on some computers.)

C. Result of comparison

1. AND

   a. TRUE if both conditions are true

   b. FALSE if either condition is false

   Examples:

   IF A = 1 AND C = 2 THEN PRINT A
   (Both conditions must be true)

   IF A > 100 AND A <= 200 THEN 20

   (If A is greater than 100 and A is less than or equal to 200, then go to statement 20)

2. OR

   a. TRUE if either condition is true

   b. FALSE if both conditions are false

   Examples:

   IF A = 1 OR C = 2 THEN PRINT A
   (Either A = 1 or C = 2 make condition true)

   IF Y$ = "YES" OR Y$ = "Y" THEN 10
   (Either Y$ = "YES" or Y$ = "Y" make condition true)

   IF A <= B OR A > C THEN 200
   (If A is less than or equal to B, or A is greater than C, then go to statement 200)

D. Order of hierarchy in comparing a condition

1. Arithmetic operation (+,-,*,/,**)

INFORMATION SHEET

2. Relational operators (=,<,>,>=,<=,<>)

3. Logical operators (AND, OR)

   a. Evaluated left to right

   b. Parentheses come first

   Example:

```
10 LET A$ = "YES"
20 LET X = 1
30 LET Y = 4
40 IF X + 1 = 4 OR Y - 1 = 3 AND A$ = "YES"
THEN PRINT "CONDITION OK"
99 END
```

delete

Step 1

Solves expression (arithmetic operation)

```
40 IF X + 1 = 4 OR Y - 1 = 3 AND A$ = "YES" THEN
      -----        -----
      1 + 1        4 - 1

        2            3
```

Step 2

Compares expressions at both sides of relational operation

```
Does 2 = 4          FALSE
Does 3 = 3          TRUE
Does A$ = "YES"     TRUE
```

Step 3

Logical evaluation of three simple conditions

```
40 IF X + 1 = 4 OR Y - 1 = 3 AND A$ = "YES" THEN

      FALSE         TRUE                TRUE

      FALSE   OR    TRUE      AND       TRUE
           \ yields /                   /
            \ TRUE /                   /
               \               yields /
                 \______________ TRUE /
```

INFORMATION SHEET

VI. Counters

A. Purpose

1. To make a count of events in order to report the total number of occurrences

2. To keep track of the number of times an event has occurred so that some action may be taken when the count reaches a certain value

B. Setting up counters

1. Assign variable name to counter

2. Initialize value

```
10 LET C = 1
20 LET T = 0
```

3. Code expressions to accumulate value of counter ~~or incorrect counte~~r

```
20 LET C = C + 1
30 LET T = T + N
```

(NOTE: Increment may be any number, a fraction of a number, or a variable (may be negative or positive).)

Example of a program using a counter:

```
100 REM  THIS PROGRAM IS USED TO COMPUTE THE
110 REM  AVERAGE GRADE
120 REM  T = TOTAL OF GRADES
130 REM  C = COUNTER FOR NUMBER OF VALUES TO BE
140 REM      AVERAGED
150 REM      A = AVERAGE
160 REM      G = GRADES
170 REM
180 REM      A = T/C
190 REM
200 LET T = 0
210 LET C = 0
220 READ G
230 LET T = T + G
240 LET C = C + 1
250 IF C = 8 THEN 300
260 GO TO 220
300 LET A = T/C
310 PRINT "AVERAGE =";A
400 DATA 100,58,95,99,87,71,94,88
999 END
```

INFORMATION SHEET

C. The total of the counter may be used in

1. Calculations of program

   Example: To compute average grades as shown in previous example

2. Control of looping

   a. Initialize counter

      `10 LET K = 1`

   b. Increment by steps

      `K = K + 1 or K = K + 2 etc.`

   c. Test for completion of loop

      `IF K = 10 THEN 999`

      (NOTE: Can increment by positive, negative, real, and integer values.)

      Example:

```
100 REM  C = COUNTER
110 REM  T = TOTAL PAY
120 REM  W = WEEKLY PAY
130 REM  LOOPING AND COUNTERS
140 PRINT "WEEK NO.","WEEKLY PAY","TOTAL PAY"
150 LET C = 0
160 LET T = 0
170 LET W = 275
180 LET T = T + W
190 LET C = C + 1
200 PRINT C,W,T
210 IF C = 4 THEN 999
220 GO TO 180
999 END
```

```
RUN

WEEK NO.        WEEKLY PAY      TOTAL PAY
1               275             275
2               275             550
3               275             825
4               275             1100
```

INFORMATION SHEET

3. Printing out counter value on terminal

```
100 PRINT "WEEK NO.","SALES","PURCHASE"
110 PRINT
120 LET S1 = 0
130 LET P1 = 0
140 LET K = 1
150 READ S,P
160 LET S1 = S1 + S
170 LET P1 = P1 + P
180 PRINT K,S,P
190 LET K = K + 1
200 IF K > 5 THEN 220
210 GO TO 150
220 PRINT "TOTAL SALES =";S1
230 PRINT "TOTAL PURCHASES =";P1
300 DATA 5000,2000,7000,4000,10000
310 DATA 6000,3000,1000,7000,8000
999 END

RUN

WEEK NO.       SALES          PURCHASES

1              5000           2000
2              7000           4000
3              10000          6000
4              3000           1000
5              7000           8000
TOTAL SALES = 32000
TOTAL PURCHASES = 21000
```

PARTIAL COLLATING SEQUENCE OF ASCII CODE

(BLANK) LOW
!
::
#
$
%
&
'
(
)
*
+
,
-
.
/
0
1
2 →
3 HIGH

4 LOW
5
6
7
8
9
:
;
<
=
>
?
@
A
B
C
D
E →
F
G HIGH

H LOW
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z HIGH

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

ASSIGNMENT SHEET #1--Write appropriate relational operators to correctly complete expressions

Directions: Write the correct relations or symbols to complete the following expessions.

Example:

3 ___<___ 4   3 ___<=___ 4   3 ___<>___ 4

1. 3 + 2 _____ 6 + 1   3 + 2 _____ 6 + 1   3 + 2 _____ 6 + 1
2. 2 * 2 _____ 4   2 * 2 _____ 4   2 * 2 _____ 4
3. 2 + 2 _____ 5 + 2   2 + 2 _____ 5 + 2   2 + 2 _____ 5 + 2
4. 6 - 3 _____ 10 - 7   6 - 3 _____ 10 - 7   6 - 3 _____ 10 - 7
5. 5 * 2 _____ 5 * 1   5 * 2 _____ 5 * 1   5 * 2 _____ 5 * 1
6. "ABC" _____ "ABCDE"   "ABC" _____ "ABCDE"   "ABC" _____ "ABCDE"

"AGE"   "ATE"

7. ~~"ATE"~~ _____ "AGE"   "AGE" _____ "ATE"   "AGE" _____ "ATE"
8. A = 6
B = 9

A _____ B   A _____ B   A _____ B

9. A = 3
B = 12

B/A _____ A + B   B/A _____ A + B   B/A _____ A + B

10. A$ = "YES"

A$ _____ "NO"   A$ _____ "NO"   A$ _____ "NO"

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

ASSIGNMENT SHEET #2--Write statements to transfer control using the GO TO statement and the IF THEN statement

Directions: Using GO TO statements and IF THEN statements, write statements to transfer the control of a program for the following conditions.

1. Write a single statement to transfer control from line number 150 to line number 100.

   ____

2. Write a statement to transfer control from line 20 to line 999.

   ____

3. Write a statement to conditionally branch from line 90 to line 70 if X is nonnegative.

   ____

4. Write a statement to conditionally branch from line 120 to line 210 if the value stored in T is not zero.

   ____

5. Write a statement to conditionally branch from line 20 to line 100 when the value stored in T is equal to the value stored in Q.

   ____

6. Write a statement to conditionally branch from line 20 to line 100 when the value stored in T is smaller than twice the value stored in Q.

   ____

7. Write a statement to conditionally branch from line 20 to line 100 when the value stored in N is equal to 30.

   ____

ASSIGNMENT SHEET #2

8. Write a statement to conditionally branch from line 20 to line 30 when the value stored in A is equal to the value stored in B and A is less than the value stored in C.

____

9. If A is greater than or equal to B, go to statement number 40 from line 90.

____

10. If A is not equal to B, go to statement on line 10 from line 50.

____

11. If A is equal to B and greater than or equal to D, go to statement at line 20 from line 70.

____

12. If A is greater than C, go to statement number 10; if not, go to statement number 30 from line 100.

____

____

13. If A is less than B, go to line 30; if A is equal to C, go to line 40; if A is greater than B, go to line 50 from line 200.

____

____

____

14. If A is equal to B, go to line number 60; if not, go to 20 from 200.

____

____

15. At line 400, if A is equal to B, Print "A = "B"; if not, print "A DOES NOT EQUAL B".

____

____

ASSIGNMENT SHEET #2

16. At line 10, if A$ is equal to the word YES, terminate the program at line 999; if A$ is not equal to the word YES, continue with the program.

    ____

17. At line 20, if A$ is equal to the word "NO" or the letter "N", continue with the program; if A$ is not equal to the word NO or the letter N, terminate the program at line 999.

    ____

18. At line 100, if A$ is equal to the word "FEMALE", add one to the counter F.

    ____

19. At line 100, if A$ is equal to the word "MALE", add one to the counter M.

    ____

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

ASSIGNMENT SHEET #3--Code, enter, and RUN programs using GO TO statements, IF THEN statements, logical operators, and counters

Directions: Code, enter, and RUN programs for the following problems.

1. Code, enter, and RUN a program to increase the variable C from 10 to 100 by 10's. Print the numbers in one vertical column.

2. Revise problem 1 to add two extra columns - one to count the number of times the loop is completed, and one to accumulate the totals (sums of the amounts). The output will be similar to the following:

```
Loop Number          Amounts          Totals

1                    10               10
2                    20               30
```

3. Code, enter, and RUN a program to input a person's height in feet and inches using INPUT statements with descriptive messages.

   Convert feet to inches and total inches to centimeters.

   1 foot = 12 inches
   1 inch = 2.54 centimeters

   Use a flag to end the program.

   Example:

```
100 PRINT "DO YOU WISH TO ENTER ANOTHER PERSON'S HEIGHT?"
110 INPUT "TYPE YES OR NO";A$
120 IF A$ = "YES" THEN 260
```

4. Code, enter, and RUN a program to read 20 numbers and to count the numbers as positive, zero, or negative.

   Find the sum of the positive numbers and the sum of the negative numbers.

   Use the following data: 10, -10, 20, -30, 50, -20, 5, 0, -7, -3, 100, -200, 65, -14, -6, 16, -29, 28, -77, 44

   Use a counter to terminate the program after the 20 numbers are processed.

ASSIGNMENT SHEET #3

5. The San Diego Car Rental Agency rents cars and charges a rate based on the number of days the car is rented.

1 day	$30 per day
2 - 5 days	28 per day
6 - 10 days	22 per day
11 - 15 days	20 per day
over 15 days	18 per day

Compute the amount charged for the following customers:

J. Jay	13 days
A. Swartz	2 days
R. Hart	1 day
T. Haynie	27 days
D. Burns	7 days
J. Wick	15 days

Print out the name, days rented, amount per day, and total amount charged.

Use the flag "EXIT" to end the program.

6. Code, enter, and RUN a program to select exam scores in the following categories.

The grades must be in a range from 1 to 100 or an error message will print "NOT A VALID GRADE".

90 - 100	A
80 - 89	B
70 - 79	C
60 - 69	D
1 - 59	F

Print the total number of grades in each category.

Grades are: 100, 75, 60, 24, -3, 82, 96, 69, 55, 103, 89, 73, 62, 60, 59, 96, 100, 23, 75, 84

Use a flag at the end of the data to exit from the loop so the total can be printed.

7. Code, enter, and RUN a program to READ 10 numbers and then print the high and the low number.

ASSIGNMENT SHEET #3

8. Code, enter, and RUN a program to INPUT 2 names at a time and arrange them in alphabetic order.

The output should resemble:

JOHNSON COMES BEFORE WILSON

Use the following names:

SMITH	JONES
JOHNSON	BROWN
THOMPSON	THOMAS
MARTIN	MORRISON
YOUR NAME	A NAME OF YOUR FRIEND

Use an INPUT statement to end the program similar to the one used in program 1.

9. Code, enter, and RUN a program to select a list of male students in age categories between 20 and 40.

Use the following data:

ANN SMITH	FEMALE	18
BOB SMITH	MALE	26
TIM JONES	MALE	16
ROBERT JOHNSON	MALE	56
ALICE BROWNE	FEMALE	24
ALICIA ABBOTT	FEMALE	25
BEN CARR	MALE	20
CAROL DRUM	FEMALE	63
LAST NAME	N	0

10. Code, enter, and RUN a program to search data for a selected discontinued part number. Use a RESTORE statement to repeat the search as many times as desired.

An INPUT statement will be needed to enter the value to be searched for. Another INPUT statement will be needed to determine if the program will end or will begin the search again. Use a statement similar to "DO YOU WISH TO SEARCH FOR ANOTHER NUMBER".

Use the following data for discontinued part numbers:

1442,1676,1924,2106,2107,2390,2802,2092,2903,2904
3000,3001,3003,3006,3305,3335,3535,3536,3690,3691

Use a flag to signal end of data.

ASSIGNMENT SHEET #3

If the number is found, print a message similar to:

PART NUMBER 1442 HAS BEEN DISCONTINUED

If the number is not found, print a message similar to:

PART NUMBER 1440 IS AVAILABLE

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

1. <   <=   <>
2. =   >=   <=
3. <   <=   <>
4. =   >=   <=
5. >   >=   <>
6. <   <=   <>
7. >   >=   <>
8. <   <=   <>
9. <   <=   <>
10. <   <>   <=

Assignment Sheet #2

```
 1.  150 GO TO 100
 2.  20 GO TO 999
 3.  90 IF X >= 0 THEN 70
 4.  120 IF T <> 0 THEN 210
 5.  20 IF T = Q THEN 100
 6.  20 IF T < Q * 2 THEN 100
 7.  20 IF N = 30 THEN 100
 8.  20 IF A = B AND A < C THEN 30
 9.  90 IF A >= B THEN 40
10.  50 IF A <> B THEN 10
11.  70 IF A = B AND A >= D THEN 20
12.  100 IF A > C THEN 10
     110 GO TO 30
```

```
13.   200 IF A < B THEN 30
      210 IF A = C THEN 40
      220 IF A > B THEN 50

14.   200 IF A = B THEN 60
      210 GO TO 20

15.   400 IF A = B THEN PRINT "A = B"
      410 PRINT "A DOES NOT EQUAL B"

16.   10 IF A$ = "YES" THEN 999

17.   20 IF A$ <> "NO" OR A$ <> "N" THEN 999

18.   100 IF A$ = "FEMALE" THEN F = F + 1

19.   100 IF A$ = "MALE" THEN M = M + 1
```

Assignment Sheet #3

```
 1. 10 C = 10
    20 PRINT C
    30 C = C + 10
    40 IF C <=100 THEN 20
    99 END

 2. 10 PRINT "LOOP NO.","AMOUNT","TOTALS"
    15 K = 1
    20 C = 10
    30 S = C
    40 PRINT K,C,S
    50 C = C + 10
    60 S = S + C
    70 IF C <= 100 THEN 40
    99 END

 3. 100 REM  THIS PROGRAM USES INPUT STATEMENTS TO OBTAIN DATA
    110 REM  THAT WILL BE CONVERTED FROM FEET AND INCHES
    120 REM  TO CENTIMETERS
    130 REM
    140 REM     1 FOOT = 12 INCHES
    150 REM     1 INCH = 2.54 CENTIMETERS
    160 REM     A$ = ANSWER TO QUESTION TO BEGIN LOOP AGAIN
    170 REM     F  = FEET
    180 REM     I  = INCHES
    185 REM     I1 = TOTAL INCHES (AFTER FEET HAS BEEN CONVERTED)
    190 REM     C  = CENTIMETERS
    195 REM
    200 INPUT "ENTER HEIGHT IN FEET AND INCHES WITH COMMAS
    BETWEEN",F,I
    210 T1 = F * 12 + I
    220 C = T1 * 2.54
    230 PRINT "A PERSON";F;"FEET";I;"INCHES IS ";C;"CENTIMETERS TALL"
    240 PRINT
```

```
250 PRINT "DO YOU WISH TO ENTER ANOTHER PERSON'S HEIGHT?"
260 INPUT "TYPE YES OR NO";A$
270 IF A$ = "YES"THEN 200
999 END
```

4.
```
100 REM  THIS PROGRAM READS 20 NUMBERS AND COUNTS THE NUMBER
110 REM  OF POSITIVE, ZERO, AND NEGATIVE NUMBERS
120 REM
140 REM  THE SUMS OF THE POSITIVE NUMBERS AND THE NEGATIVE
150 REM  NUMBERS ARE THEN CALCULATED
160 REM
170 REM     C = COUNTER
171 REM     A = NUMBER
173 REM     P = POSITIVE NUMBER
174 REM     Z = ZERO NUMBER
176 REM     N = NEGATIVE NUMBER
178 REM     P1 = TOTAL POSITIVE NUMBERS
180 REM     N1 = TOTAL NEGATIVE NUMBERS
182 REM     P2 = NUMBER OF POSITIVE NUMBERS
184 REM     N2 = NUMBER OF NEGATIVE NUMBERS
190 REM
200 C = 1
202 P1 = 0:P2 = 0
204 N1 = 0:N2 = 0
206 Z = 0
210 READ A
220 IF A = 0 THEN 300
230 IF A < 0 THEN 270
240 P1 = P1 + A
250 P2 = P2 + 1
260 PRINT A;"IS A POSITIVE NUMBER"
265 GOTO 400
270 N1 = N1 + A
280 N2 = N2 + 1
290 PRINT A;"IS A NEGATIVE NUMBER"
295 GOTO 400
300 Z = Z + 1
310 PRINT A;"IS EQUAL TO ZERO"
400 C = C + 1
410 IF C <= 20 THEN 210
420 PRINT "THE TOTAL OF THE";P2;"POSITIVE NUMBERS IS";P1
430 PRINT "THE TOTAL OF THE";N2;"NEGATIVE NUMBERS IS ";N1
440 PRINT "THERE WERE";Z;"ZERO NUMBERS"
800 DATA 10,-10,20,-30,50,-20,5,0,-7,-3,100
810 DATA -200,65,-14,-6,16,-29,28,-77,44
999 END
```

5.
```
100 REM  THIS PROGRAM CALCULATES CHARGES FOR RENTING CARS
110 REM
120 REM  THE RATES ARE:
125 REM
130 REM     1 DAY          $30
140 REM     2-5 DAYS        28
150 REM     6-10 DAYS       22
```

```
160 REM    11-15 DAYS        20
170 REM    OVER 15 DAYS      18
175 REM
180 REM     N$ = NAME OF PERSON RENTING CAR
185 REM     D  = DAYS RENTED
190 REM     A  = AMOUNT PER DAY
195 REM     T = TOTAL AMOUNT CHARGED
199 PRINT
200 PRINT "NAME","DAYS RENTED","DAILY CHARGE","TOTAL CHARGE"
201 PRINT
205 READ N$,D
210 IF N$ = "EXIT" THEN 999
220 IF D = 1 THEN A = 30:GOTO 300
230 IF D <= 5 THEN A = 28: GOTO 300
240 IF D <= 10 THEN A = 22:GOTO 300
250 IF D <= 15 THEN A = 20: GOTO 300
260 A = 18
300 T = D * A
310 PRINT N$,D,A,T
320 GOTO 205
800 DATA "J. JAY",13,"A. SWARTZ",2,"R. HART",1,"T. HAYNIE",27
810 DATA "D. BURNS",7,"J. WICK",15
820 DATA "EXIT",0
999 END
```

6.
```
100 REM  THIS PROGRAM IS USED TO ACCUMULATE THE NUMBER OF A, B,
110 REM  C, D, AND F GRADES.  GRADES LESS THAN 0 OR GREATER THAN
120 REM  100 ARE NOT INCLUDED IN THE TOTALS - AN ERROR MESSAGE IS
130 REM  PRINTED IF THE GRADE IS NOT VALID.
135 REM
140 REM  THE CLASSIFICATIONS ARE:
150 REM
160 REM       90 - 100 A
165 REM       80 -  89 B
170 REM       70 -  79 C
175 REM       60 -  69 D
180 REM        1 -  59 F
185 REM
190 N=0:B=0:C=0:D=0:F=0
200 READ G
210 IF G = -999 THEN 500
220 IF G < 0 OR G > 100 THEN 400
230 IF G >= 90 THEN A = A + 1: GOTO 200
240 IF G >= 80 THEN B = B + 1: GOTO 200
250 IF G >= 70 THEN C = C + 1: GOTO 200
260 IF G >= 60 THEN D = D + 1: GOTO 200
270 F = F + 1
280 GOTO 200
400 PRINT
410 PRINT G;"IS NOT A VALID GRADE"
420 PRINT
430 GOTO 200
```

```
   500 PRINT "GRADE", "TOTAL NUMBER"
   510 PRINT "A",A
   520 PRINT "B",B
   530 PRINT "C",C
   540 PRINT "D",D
   550 PRINT "F",F
   800 DATA 100,75,60,24,-3,82,96,69,55,103,89,73,62,60,59
   810 DATA 96,100,23,75,84,-999
   999 END
```

```
7. 100 REM  THIS PROGRAM IS USED TO READ 10 NUMBERS AND SELECT
   110 REM  THE HIGH AND THE LOW NUMBER
   120 REM
   130 REM  C = COUNTER
   140 REM  B = INITIAL VALUE READ
   150 REM  H = HIGH NUMBER
   155 REM  L = LOW NUMBER
   160 REM  N = NUMBER
   165 C = 1
   170 READ B
   175 H = B
   180 L = B
   200 READ N
   210 IF N > = H THEN H = N
   220 IF N < = L THEN L = N
   230 C = C + 1
   240 IF C <= 9 THEN 200
   250 PRINT "THE HIGHEST NUMBER IS "; H
   260 PRINT "THE LOWEST NUMBER IS "; L
   800 DATA 0,-1,2,5,1,5,555,-10,2,3,4,5,6,6
   999 END
```

```
8. 100 REM  THIS PROGRAM ARRANGES TWO NAMES IN ALPHABETIC ORDER
   110 REM
   120 REM     N1$ = NAME OF FIRST PERSON
   130 REM     N2$ = NAME OF SECOND PERSON
   140 REM      A$ = ANSWER TO QUESTION TO CONTINUE LOOP
   150 REM
   200 PRINT "TYPE IN TWO NAMES WITH QUOTATION MARKS AROUND THEM"
   210 INPUT "PLACE COMMAS BETWEEN THE NAMES";N1$,N2$
   212 PRINT
   215 IF N1$ = N2$ THEN PRINT N1$; " AND ";N2$ " ARE THE SAME ":GOTO
   400
   230 PRINT N2$; " COMES BEFORE "; N1$
   240 PRINT
   400 INPUT "DO YOU WISH TO CONTINUE ?  TYPE YES OR NO",A$
   410 IF A$ = "YES" THEN 200
   999 END
```

```
9. 100 REM  THIS PROGRAM SELECTS MALE STUDENTS BETWEEN THE
   110 REM  AGES OF 20 - 40
   115 REM
   120 REM  N$ = NAME
   130 REM  S$ = SEX
```

```
140 REM  A  = AGE
150 REM
200 READ N$,S$,A
205 IF N$ = "LAST NAME" THEN 999
210 IF S$ = "MALE" THEN GOTO 300
220 GOTO 200
300 IF A >= 20 AND A <= 40 THEN PRINT N$ ; " A MALE STUDENT
BETWEEN 20 AND 40"
310 GO TO 200
800 DATA "ANN SMITH","FEMALE",18,"BOB SMITH","MALE",26
810 DATA "TIM JONES","MALE",16,"ROBERT JOHNSON","MALE",56
820 DATA "ALICE BROWNE","FEMALE",25,"BEN CARR","MALE",20
830 DATA "CAROL DRUM","FEMALE",63,"LAST NAME","N",0
999 END
```

10.

```
010 REM  THIS PROGRAM IS USED TO SEARCH FOR DISCONTINUED PART
020 REM  NUMBERS.
030 REM
040 REM    N = PART NUMBER TO BE LOCATED
050 REM    A = DISCONTINUED PART NUMBERS
060 REM   A$ = ANSWER TO QUESTION TO CONTINUE LOOP
070 REM
100 INPUT "ENTER PART NUMBER TO BE LOCATED",N
105 PRINT
110 READ A
120 IF A = -999 THEN 200
130 IF A = N THEN PRINT "PART NUMBER";N;"HAS BEEN DISCONTINUED":
GOTO 210
140 GOTO 110
199 PRINT
200 PRINT "PART NUMBER";N;"IS AVAILABLE"
210 PRINT
215 INPUT "DO YOU WISH TO SEARCH AGAIN";A$
217 PRINT
220 IF A$ = "YES" OR A$ = "Y" THEN RESTORE:GOTO 100
230 GOTO 999
800 DATA 1442,1676,1924,2106,2107,2390,2802,2092,2903,2904
810 DATA 3000,3001,3003,3006,3305,3335,3535,3536,3690,3691
820 DATA -999
999 END
```

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS
UNIT V

NAME_________________________

TEST

1. Match the terms on the right with the correct definitions:

_____a. A comparison of two numeric or string expressions that uses a relational operator that is evaluated as either TRUE or FALSE

_____b. BASIC statement that gives line number of the statement to which an unconditional branch is made

_____c. A transfer of program control that occurs only if specified conditions are met

_____d. Symbols used to make comparison of BASIC variables and expressions, and to test relationships in conditional statements, such as equal to (=)

_____e. To alter the value of a counter by a specified value

_____f. A number or string placed at the end of a DATA statement to signal the end of data

_____g. Conditional statement used to make certain transfers and to direct further processing based upon results of the test

_____h. Used to make logical comparison with IF THEN statements for two or more relations

_____i. Statements that instruct the computer to branch to a specific statement number of an executable statement regardless of any condition

1. Conditional branching
2. Counters
3. Flag
4. GO TO statement
5. IF THEN statement
6. Increment
7. Initializing
8. Logical operator
9. Loop
10. Looping
11. Relational expression
12. Relational operator
13. Unconditional branching

_____j. Setting a variable to a specific starting value such as T=0 or C=1

_____k. The process of repeating a series of statements in a program

_____l. Sequence of instructions that is repeated until terminating condition is reached

_____m. Variable locations that are used to count the number of times a certain event occurs (loop is completed)

2. Directions: Match the following numeric and string meanings with their correct symbol by placing the correct letter in the corresponding blank.

_____a. <>
_____b. <
_____c. =
_____d. >
_____e. <=
_____f. >=

1. equal to
2. greater than or equal to
3. greater than
4. less than or equal to
5. less than
6. not equal to

_____a. <>
_____b. <
_____c. =
_____d. >
_____e. <=
_____f. >=

1. precedes
2. follows
3. equal
4. precedes or equal to
5. does not equal
6. follows or equal to

3. Select true statements concerning the GO TO statements by placing an "X" in the appropriate blanks.

_____a. Can cause transfer to the END statement

_____b. Is a conditional branching statement

_____c. The form GO TO is usually not allowed on most computers _____d. Transfers program to the line number specified

_____e. Stops the sequential execution of a program

_____f. The following is a valid GO TO statement:

```
50 GOTO 80
```

_____g. The following is a valid GO TO statement at line 30:

```
10 READ A
20 PRINT A
30 GO TO 15
40 DATA 1,2,3
99 END
```

_____h. The GO TO statement can only direct execution to a higher line number

_____i. The GO TO statement can create an endless loop

_____j. The GO TO statement uses relational operators to evaluate the expressions as TRUE or FALSE

4. Select true statements concerning the IF THEN statement by placing an "X" in the appropriate blanks.

_____a. The IF THEN statement is a form of conditional branching

_____b. The condition to be tested can only be expressions

_____c. The condition can only be TRUE or FALSE

_____d. The IF THEN statement can be used to solve the problem of the OUT OF DATA message by testing for a flag value in the data

_____e. If condition is TRUE, control passes to the next line number

_____f. If the condition is FALSE, control passes to the command or statement after the word THEN

_____g. The following is a valid IF THEN statement:

```
10 IF A + B > C + D THEN PRINT "GREATER"
```

_____h. The following is a valid IF THEN statement:

```
100 IF A$ = "YES" THEN 10
```

_____i. The following is a valid IF THEN statement:

100 IF A = 1,246 THEN 20

_____j. The following is a valid IF THEN statement:

10 IF A <= B THEN 60

5. Distinguish between TRUE and FALSE conditions in the following expressions containing logical operators by placing a "T" for TRUE and a "F" for FALSE in the blanks provided. For problems a - d, let A = 2 and B = 6; for problems e - i, let A$ = "YES", B$ = "Y", A = 10, and B = 6.

_____a. 100 IF A > B OR B = 6 THEN 10

_____b. 100 IF A * B <> 2 AND B <> 6 THEN 10

_____c. 20 IF A = 2 OR B = 6 THEN 100

_____d. 20 IF B < A OR A * B > 10 THEN PRINT "OK"

_____e. 10 IF A$ = "YES" AND A = 6 THEN 100

_____f. 10 IF A$ <> "YES" OR A$ = "Y" THEN 30

_____g. 10 IF A$ <> "YES" OR B$ = "N" THEN 100

_____h. 10 IF A <> 100 AND A > 0 THEN A = A + 1

_____i. 10 IF A <= B AND A$ <> "NO" THEN C = C + 1

6. List the three steps in setting up a counter

a.

b.

c.

7. Demonstrate the ability to write appropriate relational operators to correctly complete expressions.

8. Write statements to transfer control using the GO TO and the IF THEN statements.

9. Code, enter, and RUN programs using the counter, GO TO, and the IF THEN statements.

(NOTE: If objectives 7 - 9 have not been accomplished prior to the test, ask your instructor when they should be completed.)

PROGRAM CONTROL--RELATIONAL OPERATORS, GO TO STATEMENT, IF THEN STATEMENT, LOGICAL OPERATORS, AND COUNTERS

UNIT V

ANSWERS TO TEST

1. a. 11
   b. 4
   c. 1
   d. 12
   e. 6
   f. 3
   g. 5
   h. 8
   i. 13
   j. 7
   k. 10
   l. 9
   m. 2

2. a. 6
   b. 5
   c. 1
   d. 3
   e. 4
   f. 2

   a. 5
   b. 1
   c. 3
   d. 2
   e. 4
   f. 6

3. a.
   d.
   e.
   f.
   i.

4. a.
   c.
   d.
   g.
   h.
   j.

ANSWERS TO TEST

5. a. T
   b. F
   c. T
   d. T
   e. F
   f. F
   g. F
   h. T
   i. T

6. a. Initialize counter
   b. Increment by steps
   c. Test for completion of loop

7. Evaluated to the satisfaction of the instructor.

8. Evaluated to the satisfaction of the instructor.

9. Evaluated to the satisfaction of the instructor.

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

UNIT OBJECTIVE

After completion of this unit, the student should be able to select true statements concerning the program development cycle, match basic program flowcharting descriptions with their symbols, and match structured flowcharts with their basic designs. Code, enter, and RUN programs using steps outlined in flowcharts; construct flowcharts from coded programs; construct a flowchart and code, enter, and RUN the program flowcharted; and construct structured flowcharts in three basic designs. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with program development cycle and flowcharting with the correct definitions.
2. Select true statements concerning the program development cycle.
3. Match basic program flowcharting descriptions with their symbols.
4. Match structured flowcharts to their correct basic design.
5. Code, enter, and RUN programs using steps outlined in flowcharts.
6. Complete a program flowchart.
7. Construct program flowcharts from coded programs.
8. Demonstrate the ability to perform the steps in the program development cycle.
9. Construct structured flowcharts in three basic designs.

PROGRAM DEVELOPMENT CYCLE AND FLOWCHARTING
UNIT VI

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Prepare large laminated flowcharting symbols. Using selected programs, have students arrange the symbols unitl a desired flowchart is completed.

VI. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

- A. Objective sheet
- B. Information sheet
- C. Transparency Masters
  - 1. TM 1--BASIC Coding Form
  - 2. TM 2--BASIC Program Flowcharts
  - 3. TM 3--Sequence Structure Flowchart
  - 4. TM 4--Loop Structure Flowchart
  - 5. TM 5--Conditional Structure Flowchart
- D. Assignment sheets
  - 1. Assignment Sheet #1--Code, enter, and RUN programs using steps outlined in flowcharts
  - 2. Assignment Sheet #2--Complete a program flowchart
  - 3. Assignment Sheet #3--Construct program flowcharts from coded programs
  - 4. Assignment Sheet #4--Demonstrate the ability to perform the steps in the program development cycle

        5. Assignment Sheet #5--Construct structured flowcharts in three basic designs

    D. Answers to assignment sheets

    E. Test

    F. Answers to test

II. References:

Refer to Unit I references.

PROGRAM DEVELOPMENT CYCLE AND FLOWCHARTING
UNIT VI

INFORMATION SHEET

I. Terms and definitions

A. Program - step-by-step process described in terms of instructions a computer can follow

B. Flowchart - graphic representation of order of operations in data processing system or program in which symbols are used to represent processing operations and data flow

C. Flowcharting template - plastic or metallic card containing both system and program flowcharting symbols as cutout forms

D. Code - to translate and write information in a programming language

E. Coding forms - a programming aid designed to facilitate the coding of programs in a clear, concise manner with a minimum of ambiguity

F. Debug - task of detecting, locating, and eliminating mistakes or errors in a program

G. Logical errors - errors which occur when the programmer does not thoroughly understand a phase of the problem, fails to account for all situations that may arise during processing, or misinterprets some steps when coding

H. Syntax errors - error in the grammar or in the form of the programming language

I. Auxiliary storage - a supplemental means for storing data

J. Test - run program against representative data to ensure an adequacy of the program or system for regular use

K. Documentation - the illustrations and explanations that should accompany the computer printout of a "working" program that include both general information needed by the user of the program and the specific information needed by the computer operator or by the computer programmer if the program later needs updating

INFORMATION SHEET

L. Terminal symbol - represents the beginning or ending of a program

M. Input/output symbol - represents an input or output operation

N. Processing symbol - represents calculations, initializations, and other processing functions

O. Decision symbol - represents a decision, question, or comparison that determines which alternative paths are to be followed

P. Connector symbol - connects separated portions of a flowchart without drawing lines between the parts

Q. Flow direction - connects other symbols and indicates the sequence of operations

R. Offpage connector - designates exit from or entry to another page

II. Program development cycle

A. Define the problem

1. Understand the tasks to be accomplished

2. Outline needed information and data

3. Understand output required

B. Plan the solution

1. Break down problem into smaller parts

2. Prepare program flowchart

a. Show in detail the processing steps and order of execution

b. Use program flowchart symbols

(NOTE: A flowcharting template is used in preparing a program flowchart.)

C. Code the solution

1. Use BASIC statements to write out step-by-step instructions

INFORMATION SHEET

   2. Use flowchart as a guide

   3. Use BASIC coding forms (Transparency 1)

D. Review the program and debug

   1. Program examined for logical errors of the flowchart

   2. Program examined for syntax errors

   3. Review should be done by other people as well as the programmer to find errors made by the programmer

   4. Review is done before program is entered into computer

   5. Program should contain no errors after the review

   6. Less time is needed for testing after review

   (NOTE: Before using a terminal, walk through the program step by step. This is referred to as a structured walk through or desk debugging.)

E. Enter program into computer

   1. Enter program

   2. Save on auxiliary storage device

   3. Execute (RUN) program

F. Test and debug

   1. This step verifies that the program will consistently produce correct results

   2. Both valid and invalid data submitted as input

   3. Expected results determined before input

      (NOTE: This allows for the validity of the information produced from the computer to be checked.)

G. Document

      (NOTE: Clear documentation is mark of well written program.)

   1. Documentation should include

INFORMATION SHEET

a. Final flowchart

b. A listing of the program containing

    1) Name of program

    2) Date written

    3) Programmer

    4) Function of program

    5) List of variables used and what each represents

    6) Data used

    7) Process which is to occur

    8) Other parts needed for programmers to understand what the program is doing

c. External documentation

    1) A written description of the problem and program outlining what it does and special features

    2) Information needed about the program to aid in running or use of the program

    Examples of external documentation:

        Operating instructions

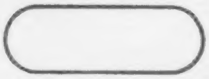
        Data format

        Error messages generated by program

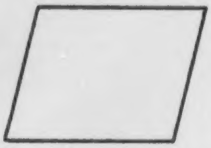
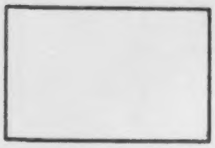
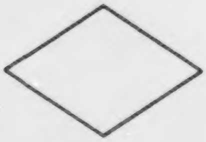
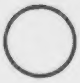
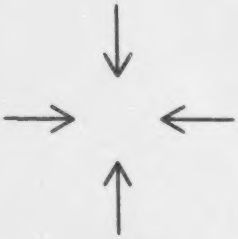
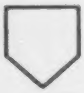
        Test cases and sample results

        Other similar information

III. BASIC program flowcharting descriptions and their symbols (Transparency 2)

A. Terminal Symbol -  Represents the beginning or end of the program

INFORMATION SHEET

B. Input/output symbol -		Represents process of putting data into storage or calling it from storage through input/output device
C. Processing symbol -		Represents calculations, initializing, and other processing functions
D. Decision symbol -		Represents a decision, question, or comparison that determines which alternate paths are to be followed
E. Connector symbol -		The notation inside this symbol signifies to which other symbol the program flow should move
F. Flow direction -		Connects other symbols and indicates the sequence of operations
G. Offpage connector -		This symbol is to designate exit from or entry to another page

INFORMATION SHEET

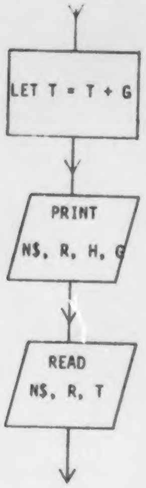
IV. Structured flowcharting (Transparencies 3, 4, and 5)

A. Flow is top down

B. Three control structures make up basic design

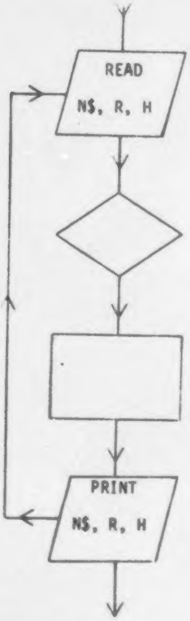
1. Sequence structure (one after another)

Example:


LET T = T + G
PRINT
N$, R, H, G
READ
N$, R, T

2. Loop structure (execution loops continue until exiting the loop is desired)

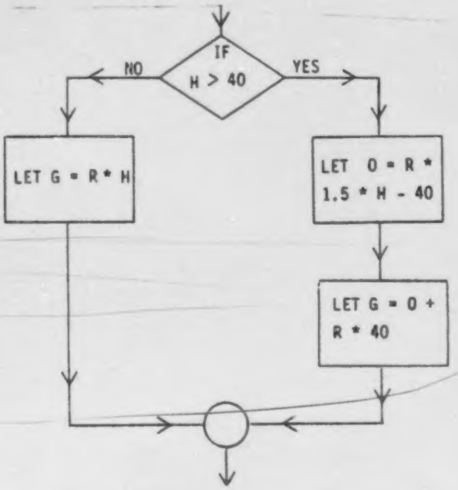
Example:


READ
N$, R, H
PRINT
N$, R, H

INFORMATION SHEET

3. Conditional structure (branching occurs depending on condition)

   Example:

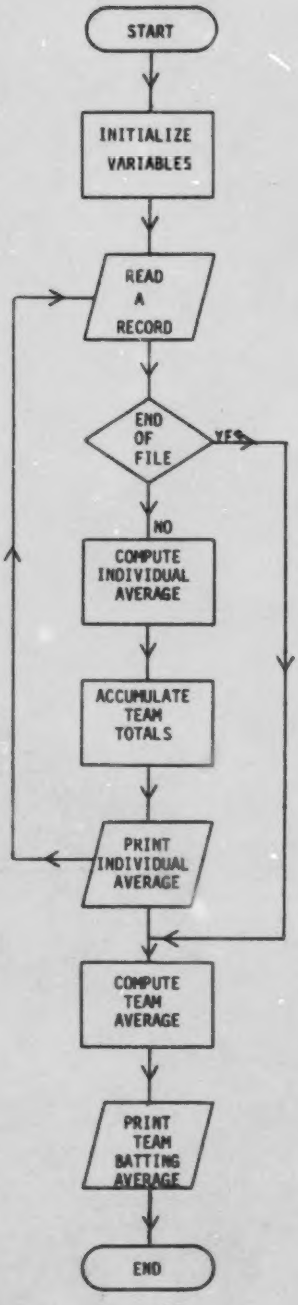

IF
H > 40
NO
YES
LET G = R * H
LET O = R * 1.5 * H - 40
LET G = O + R * 40

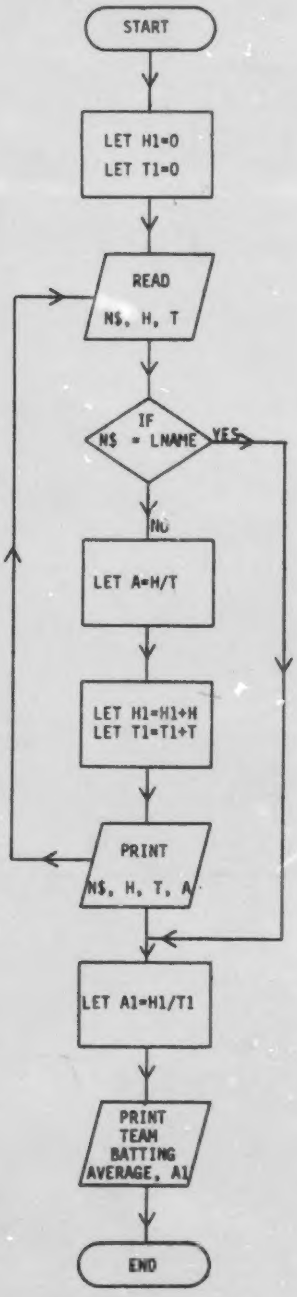
C. Each control structure has one entry and one exit

GENERAL PURPOSE TAB FORM

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80

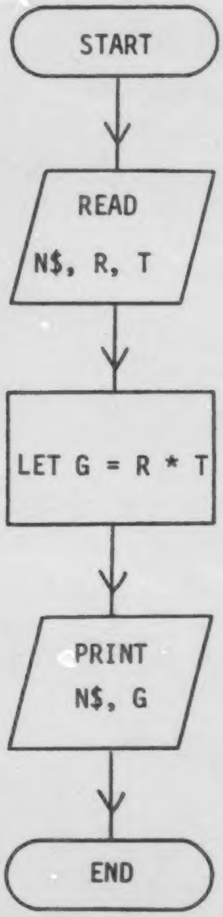
FORM NO. UTO 16

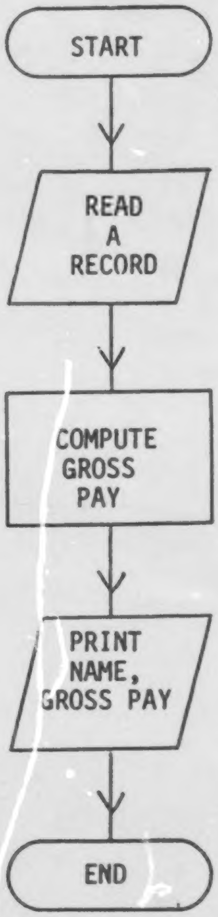

START
INITIALIZE VARIABLES
READ A RECORD
END OF FILE
YES
NO
COMPUTE INDIVIDUAL AVERAGE
ACCUMULATE TEAM TOTALS
PRINT INDIVIDUAL AVERAGE
COMPUTE TEAM AVERAGE
PRINT TEAM BATTING AVERAGE
END


START
LET H1=0
LET T1=0
READ N$, H, T
IF N$ = LNAME
YES
NO
LET A=H/T
LET H1=H1+H
LET T1=T1+T
PRINT N$, H, T, A
LET A1=H1/T1
PRINT TEAM BATTING AVERAGE, A1
END

EXAMPLE OF SEQUENCE STRUCTURE FLOWCHART

```
100 REM      THIS PROGRAM COMPUTES GROSS PAY
110 REM
120 REM      N$ = NAME
130 REM       R = RATE
140 REM       T = TIME  (HOURS WORKED)
150 REM       G = GROSS PAY
160 REM
170 REM       GROSS PAY = RATE X TIME
180 REM
200 READ N$,R,T
210 LET G = R*T
220 PRINT N$,G
300 DATA "ANDY GREEN",5, 30
999 END
```

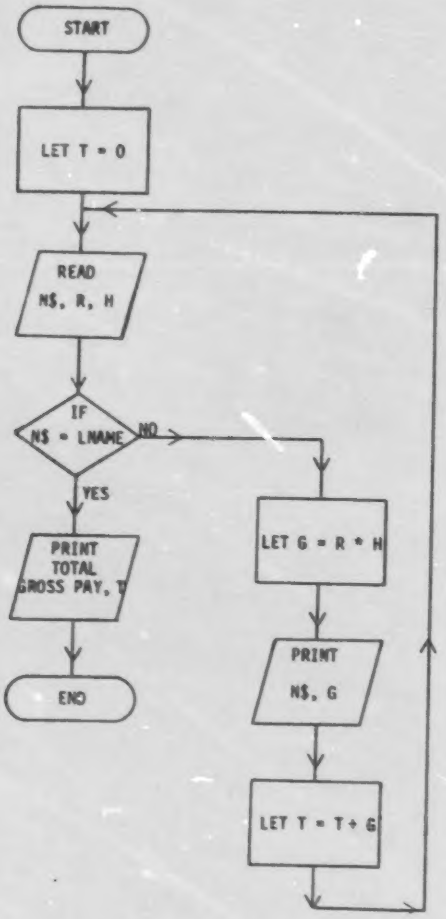

START
READ
N$, R, T
LET G = R * T
PRINT
N$, G
END

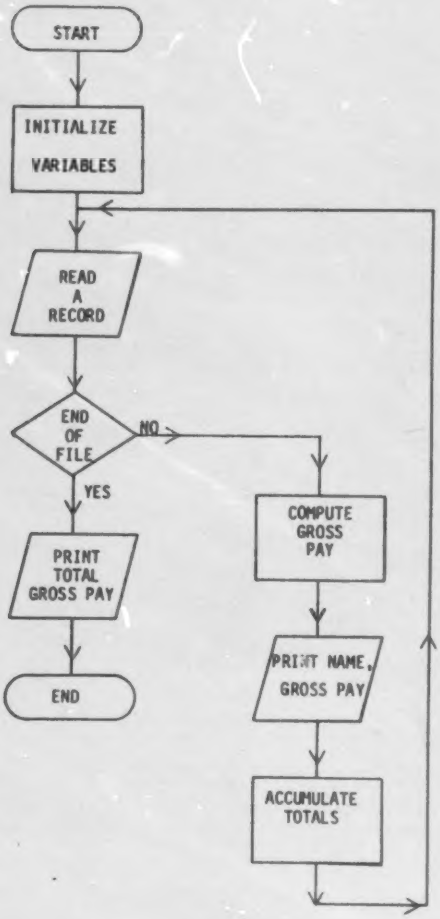

START
READ
A
RECORD
COMPUTE
GROSS
PAY
PRINT
NAME,
GROSS PAY
END

TM 3

EXAMPLE OF LOOP STRUCTURE FLOWCHART

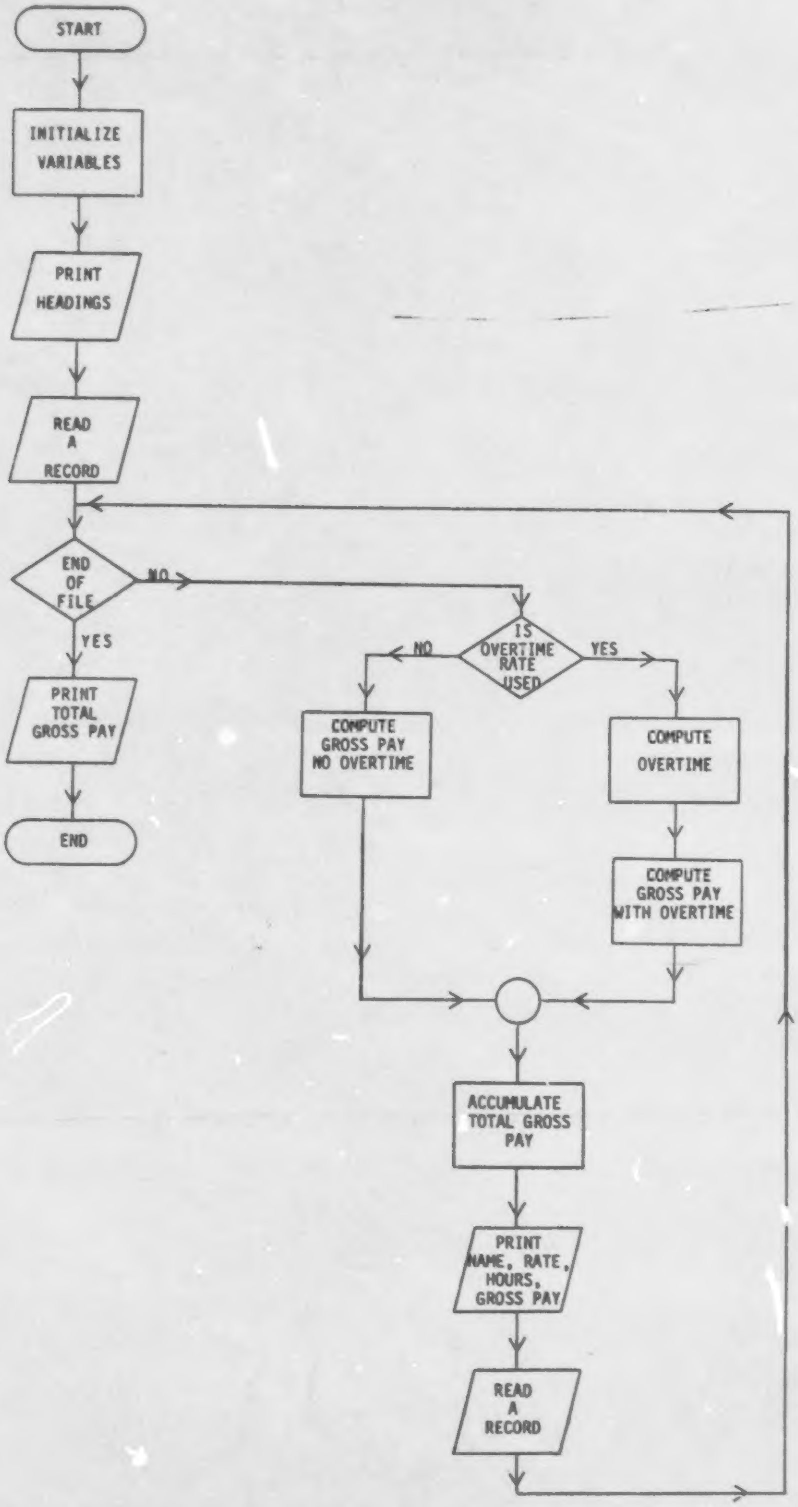
```
100 REM     THIS PROGRAM COMPUTES GROSS PAY FOR
110 REM     INDIVIDUAL EMPLOYEES AND ACCUMULATES
120 REM     THE TOTAL FOR ALL EMPLOYEES
130 REM
140 REM     N$ = NAME
150 REM      R = RATE OF PAY
160 REM      H = HOURS WORKED
170 REM      G = GROSS PAY
180 REM      T = TOTAL GROSS PAY FOR ALL EMPLOYEES
190 REM      GROSS PAY = RATE X TIME
200 LET T = 0
210 READ N$, R, H
220 IF N$ = "LNAME" GO TO 300
230 LET G = R * H
240 PRINT N$, G
250 LET T = T + G
260 GO TO 210
300 PRINT
310 PRINT "TOTAL GROSS PAY", T
800 DATA "HENRY TAYLOR",6.50,28,"MARY JONES",10.00,40,
810 DATA "MICHAEL TYSON",11.00,40,"LNAME",0,0
999 END
```


START
LET T = 0
READ N$, R, H
IF N$ = LNAME
NO
YES
PRINT TOTAL GROSS PAY, T
END
LET G = R * H
PRINT N$, G
LET T = T + G

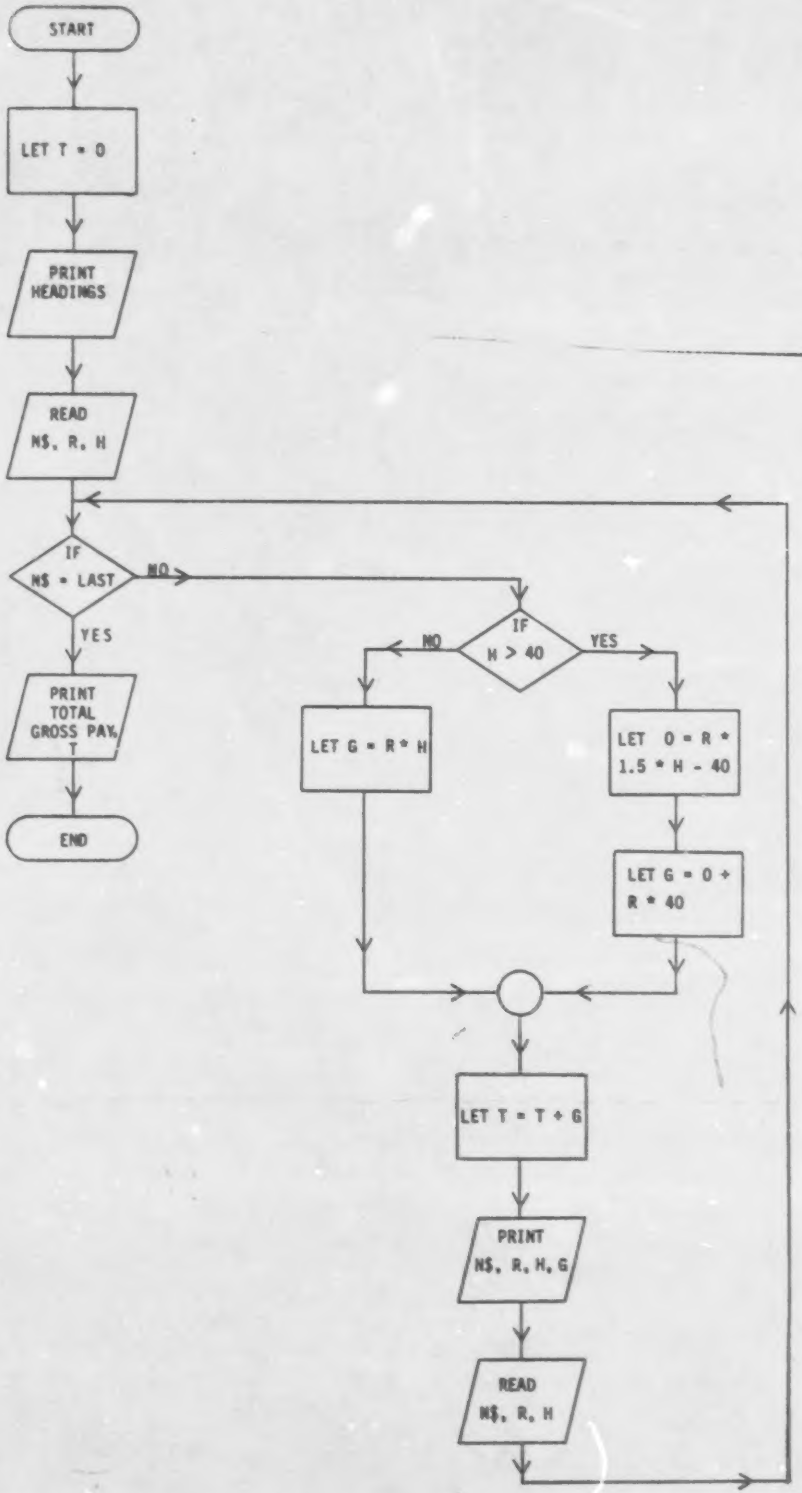

START
INITIALIZE VARIABLES
READ A RECORD
END OF FILE
NO
YES
PRINT TOTAL GROSS PAY
END
COMPUTE GROSS PAY
PRINT NAME, GROSS PAY
ACCUMULATE TOTALS

EXAMPLE OF CONDITIONAL STRUCTURE FLOWCHART

```
100 REM      THIS PROGRAM COMPUTES GROSS PAY FOR
110 REM      INDIVIDUAL EMPLOYEES.  IT ALSO
120 REM      ACCUMULATES TOTAL GROSS PAY FOR
130 REM      ALL EMPLOYEES
140 REM
150 REM                     N$ = NAME
160 REM                      R = RATE OF PAY
170 REM                      H = HOURS WORKED
180 REM                      G = GROSS PAY
190 REM                      O = OVERTIME PAY
200 REM                      T = TOTAL GROSS PAY FOR ALL EMPLOYEES
205 REM
210 REM*** CALCULATES GROSS PAY WHEN NO OVER TIME IS INVOLVED***
220 REM          GROSS PAY = RATE OF PAY X HOURS WORKED
225 REM
230 REM   *** CALCULATES GROSS PAY WHEN OVERTIME IS INVOLVED***
240 REM  OVERTIME PAY = RATE OF PAY * 1.5 FOR ALL HOURS OVER 40
250 REM       GROSS PAY = OVERTIME PAY + RATE OF PAY X 40 HOURS
260 REM
300 LET T = 0
310 PRINT "EMPLOYEES NAME", "HOURS WORKED", "GROSS PAY"
320 READ N$, R, H
330 IF N$ = "LAST" GO TO 510
340 IF H > 40 GO TO 410
345 REM          ***COMPUTES GROSS PAY WITHOUT OVERTIME***
350 LET G = R * H
360 GO TO 450
400 REM          ***COMPUTES GROSS PAY WITH OVERTIME***
410 LET O = R * 1.5 * H - 40
420 LET G = O + R * 40
450 REM          ***ACCUMULATES TOTAL GROSS PAY***
460 LET T = T + G
465 REM   ***PRINTS NAME, RATE, HOURS, GROSS PAY FOR INDIVIDUAL
470 REM      EMPLOYEES***
475 PRINT N$, R, H, G
480 READ N$, R, H
485 READ GO TO 330
500 REM          ***PRINTS TOTAL GROSS PAY FOR ALL EMPLOYEES***
510 PRINT
520 PRINT "TOTAL GROSS PAY",T
999 END
```


START
INITIALIZE VARIABLES
PRINT HEADINGS
READ A RECORD
END OF FILE
NO
YES
PRINT TOTAL GROSS PAY
END
IS OVERTIME RATE USED
NO
YES
COMPUTE GROSS PAY NO OVERTIME
COMPUTE OVERTIME
COMPUTE GROSS PAY WITH OVERTIME
ACCUMULATE TOTAL GROSS PAY
PRINT NAME, RATE, HOURS, GROSS PAY
READ A RECORD

TM 5


START
LET T = 0
PRINT HEADINGS
READ N$, R, H
IF N$ = LAST
NO
YES
PRINT TOTAL GROSS PAY, T
END
IF H > 40
NO
YES
LET G = R * H
LET O = R * 1.5 * H - 40
LET G = O + R * 40
LET T = T + G
PRINT N$, R, H, G
READ N$, R, H

PROGRAM DEVELOPMENT CYCLE AND FLOWCHARTING
UNIT VI

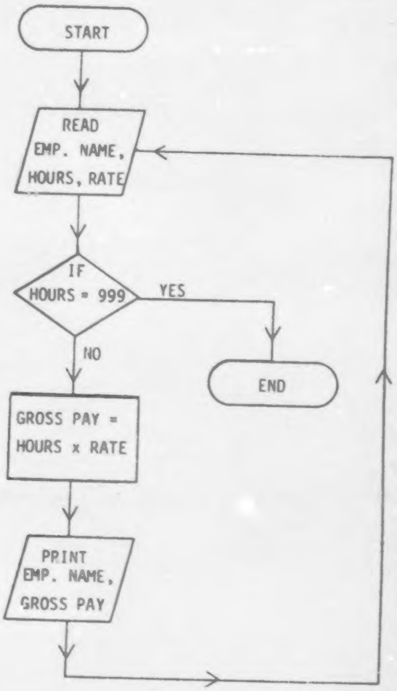
ASSIGNMENT SHEET #1--Code, enter, and RUN programs using steps in flowcharts

Directions: Code, enter, and RUN programs using the flowcharts and data listed below. Use the coding form provided with this unit.

1. Program 1

   a. Code a program using the following data and flowchart. Use the coding form provided with this unit.

Name	Hours	Rate
Abbot	40	2.50
Brown	30	3.00
Coffy	35	4.00
Smith	-999	0.00

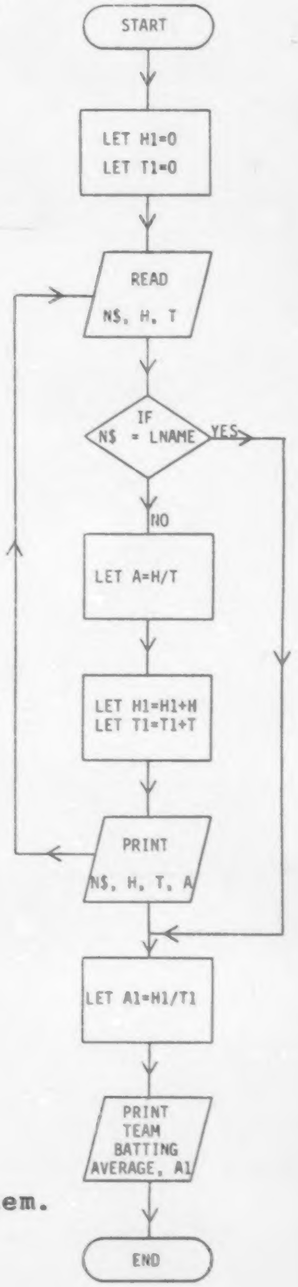

START
READ EMP. NAME, HOURS, RATE
IF HOURS = 999
YES
NO
END
GROSS PAY = HOURS x RATE
PRINT EMP. NAME, GROSS PAY

   b. Enter and RUN the program on a computer system.

2. Program 2

   a. Code a program using the following data and flowchart. Use the coding form provided with this unit.

Name	Number of Hits	Number of Times at Bat
ADAMS	1	5
BELL	3	5
CARTER	5	5
DILLON	3	5
HALEY	4	5
LOTT	0	5
SMITH	3	5
THOMPSON	2	5
WILSON	4	5


START
LET H1=0
LET T1=0
READ
N$, H, T
IF
N$ = LNAME
YES
NO
LET A=H/T
LET H1=H1+H
LET T1=T1+T
PRINT
N$, H, T, A
LET A1=H1/T1
PRINT
TEAM
BATTING
AVERAGE, A1
END

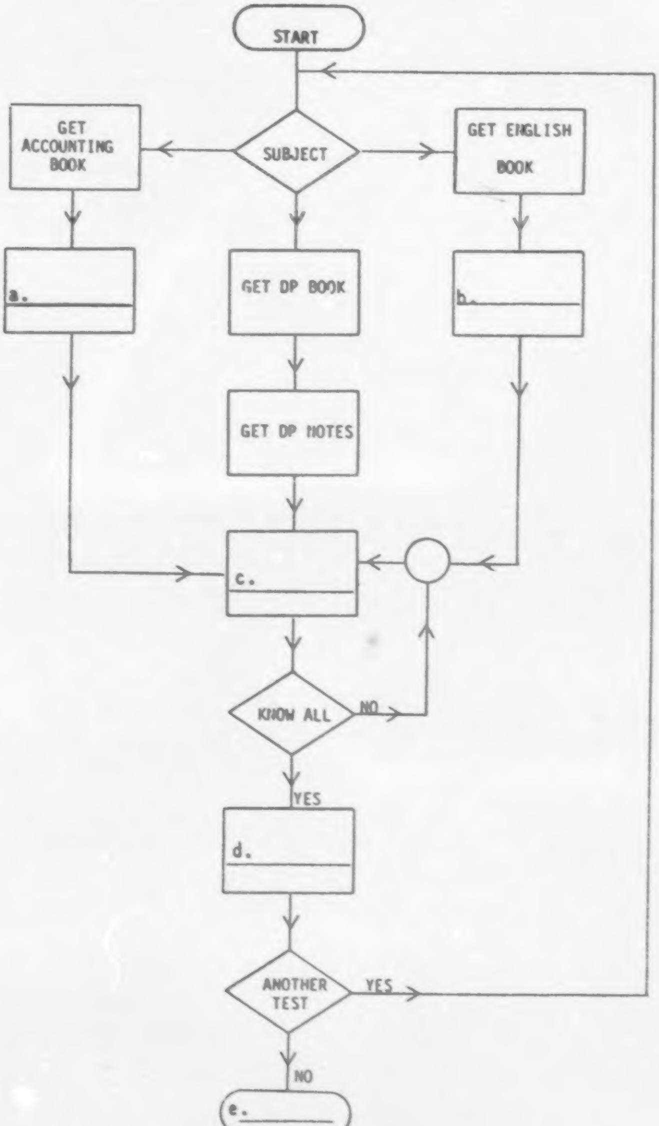
   b. Enter and RUN the program on a computer system.

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

ASSIGNMENT SHEET #2--Complete a simple program flowchart

Directions: Complete the blanks with the words listed below to show the procedure in studying for tests.

PUT AWAY MATERIALS
GET ACCOUNTING NOTES
GET ENGLISH NOTES
STUDY
END


START
SUBJECT
GET ACCOUNTING BOOK
GET ENGLISH BOOK
a. ________
GET DP BOOK
b. ________
GET DP NOTES
c. ________
KNOW ALL
NO
YES
d. ________
ANOTHER TEST
YES
NO
e. ________

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

ASSIGNMENT SHEET #3--Construct program flowcharts from coded programs

Directions: Using appropriate program flowcharting symbols, construct a flowchart from the programs listed below.

1. Program 1

```
100 REM THIS PROGRAM CALCULATES AVERAGE TEST SCORES
110 REM
120 REM        N$ = STUDENT NAME
130 REM         G = TEST GRADE
140 REM         T = TOTAL COUNTER
150 REM         N = NUMBER OF VALUES COUNTER
160 REM
170 REM        AVERAGE = TOTAL COUNTER/NUMBER OF VALUES COUNTER
180 REM
200 READ N$
210 IF N$ = "END" THEN GO TO 999
220 LET T = 0
230 LET N = 0
240 READ G
250 IF G = -999 THEN GO TO 290
260 LET T = T + G
270 LET N = N + 1
280 GO TO 240
290 REM        CALCULATES AVERAGE SCORE
300 LET A = T/N
310 REM        PRINTS NAME AND AVERAGE SCORE
320 PRINT
330 PRINT "NAME", "AVERAGE"
340 PRINT N$,A
350 GO TO 200
360 DATA "R BACON",100,80,90,100,-999
370 DATA "D BANKS",100,90,70,80,-999
380 DATA "J BLANK",90,74,100,85,-999
390 DATA "S BRYANT",80,100,75,85,-999
400 DATA "END",0,0,0,0
999 END
```

2. Program 2

```
100 REM        THIS PROGRAM COMPUTES TRIP TIME AND COST
110 REM
120 REM        D = DISTANCE
130 REM        S = AVERAGE SPEED
140 REM        T = TIME OF TRIP
150 REM        M = MILEAGE RATE
160 REM        C = COST PER GALLON
170 REM        E = ESTIMATED TRIP EXPENSE
180 REM        G = NUMBER OF GALLONS OF FUEL USED
190 REM
200 REM        TIME OF TRIP = DISTANCE/SPEED
210 REM        NUMBER OF GALLONS USED = DISTANCE /MILES PER GALLON
220 REM        ESTIMATED TRIP EXPENSE = NUMBER OF GALLONS OF FUEL
230 REM                                 USED X COST PER GALLON
240 REM
300 PRINT "PROGRAM TO COMPUTE TRIP TIME AND COST"
310 PRINT
320 READ D,S,M,C
330 IF D = -99 THEN GO TO 999
340 LET T = D/S
350 LET G = D/M
360 LET E = G * C
370 PRINT "DISTANCE = "; D; " MILES"
380 PRINT "AVERAGE SPEED = "; S; " MILES PER HOUR"
390 PRINT "TIME OF TRIP = "; T; " HOURS"
400 PRINT "MILEAGE RATE = "; M; " MILES PER GALLON"
410 PRINT "COST PER GALLON = " C; " DOLLARS"
420 PRINT
430 PRINT "ESTIMATED TRIP EXPENSE = " E; " DOLLARS"
440 GO TO 320
500 DATA 320,50,19.5,80
510 DATA 400,60,13,1.07
520 DATA 1000, 55,25,1.20
530 DATA -99,0,0,0
999 END
```

3. Program 3

```
100 REM   THIS PROGRAM COMPUTES DISCOUNTS ACCORDING TO UNITS
110 REM   ORDERED
120 REM
130 REM        N = PRODUCE NUMBER
140 REM        P = PRICE
150 REM        Q = QUANTITY
160 REM        D = DISCOUNT
170 REM
180 REM   DISCOUNTS OF 0% ARE ALLOWED FOR ORDERS OF 50 UNITS
190 REM    OR LESS
200 REM   DISCOUNTS OF 10% ARE ALLOWED FOR ORDERS OVER 50
210 REM   DISCOUNTS OF 20% ARE ALLOWED FOR ORDERS OVER 100
220 REM
230 REM   DISCOUNT = % OF DISCOUNT X PRICE X QUANTITY
240 REM
300 PRINT"PRODUCE NO.","PRICE","QUANTITY","DISCOUNTED AMOUNT"
310 PRINT
320 READ N,P,Q
330 IF N = -999 THEN 999
340 IF Q <= 50 THEN 400
350 IF P < 100 THEN 500
360 LET D = .20 * P * Q
365 PRINT N,P,Q,D
370 GO TO 320
400 PRINT N,P,Q,"NO DISCOUNT"
410 GO TO 320
500 LET D = .10 * P * Q
510 PRINT N,P,Q,D
520 GO TO 320
600 DATA 100,3,400,125,5,25
610 DATA 150,6,60,-999,0,0
999 END
```

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

ASSIGNMENT SHEET #4--Demonstrate the ability to perform the steps in the program development cycle

Directions: Using the program development cycle, design a program to read in 10 numbers into the computer, find the sum, and the average of these numbers. Refer to these six numbers as "N" each time in your flowchart. Use -999 as your check for the last number read. When the last number has been read into the computer, compute the average, print the average, and stop.

Turn in to your instructor the following:

A flowchart
A listing of the program with appropriate documentation
A copy of the RUN of the program

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

ASSIGNMENT SHEET #5--Construct structured flowcharts in three basic designs

Directions: Using appropriate program flowcharting symbols, construct three simple flowcharts in the basic designs listed below.

1. Sequence Structure

2. Loop Structure

3. Conditional Structure

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

1. Problem 1

```
LIST

100 REM
110 REM   THIS PROGRAM CALCULATES GROSS PAY
120 REM
130 REM   N$ = NAME
140 REM    H = HOURS
150 REM    R = RATE
160 REM    P = GROSS PAY
170 REM
180 REM   GROSS PAY = RATE X HOURS
190 REM
200 READ N$,H,R
210 IF H = -999 THEN 999
220 LET P = R * H
230 PRINT N$,P
240 GO TO 200
300 DATA "ABBOT",40,250,"BROWN",30,3.00,"COFFY",35,4
310 DATA "SMITH",-999,0
999 END

Ready

RUN

ABBOT       100
BROWN       90
COFFY       150

Ready
```

2. Problem 2

```
10 REM   THIS PROGRAM CALCULATES BATTING AVERAGES
20 REM
30 REM    H = NUMBER OF HITS
40 REM    T = NUMBER OF TIMES AT BAT
50 REM   N$ = NAME OF BATTER
60 REM   A1 = TEAM AVERAGE
70 REM   H1 = COUNTER FOR TEAM HITS
80 REM   T1 = COUNTER FOR TEAM TIMES AT BAT
90 REM
100 LET H1 = 0
110 LET T1 = 0
120 READ N$,H,T
130 IF N$ = "LAST NAME" THEN GO TO 200
140 LET A = H/T
150 LET H1 = H1 + H
160 LET T1 = T1 + T
170 PRINT N$,H,T,A
180 GO TO 120
200 LET A1 = H1/T1
210 PRINT
220 PRINT "TEAM BATTING AVERAGE",A1
230 DATA "ADAMS",1,5,"BELL",3,5,"CARTER",5,5
240 DATA "DILLON",3,5,"HALEY",4,5,"LOTT",0,5
250 DATA "SMITH",3,5,"THOMPSON",2,5,"WILSON",4,5
260 DATA "LAST NAME",0,0
999 END

Ready

RUN

ADAMS           1             5             .2
BELL            3             5             .6
CARTER          5             5             1
DILLON          3             5             .6
HALEY           4             5             .8
LOTT            0             5             0
SMITH           3             5             .6
THOMPSON        2             5             .4
WILSON          4             5             .8

TEAM BATTING AVERAGE          .555556
```

Assignment Sheet #2

a. Get accounting notes

b. Get English notes

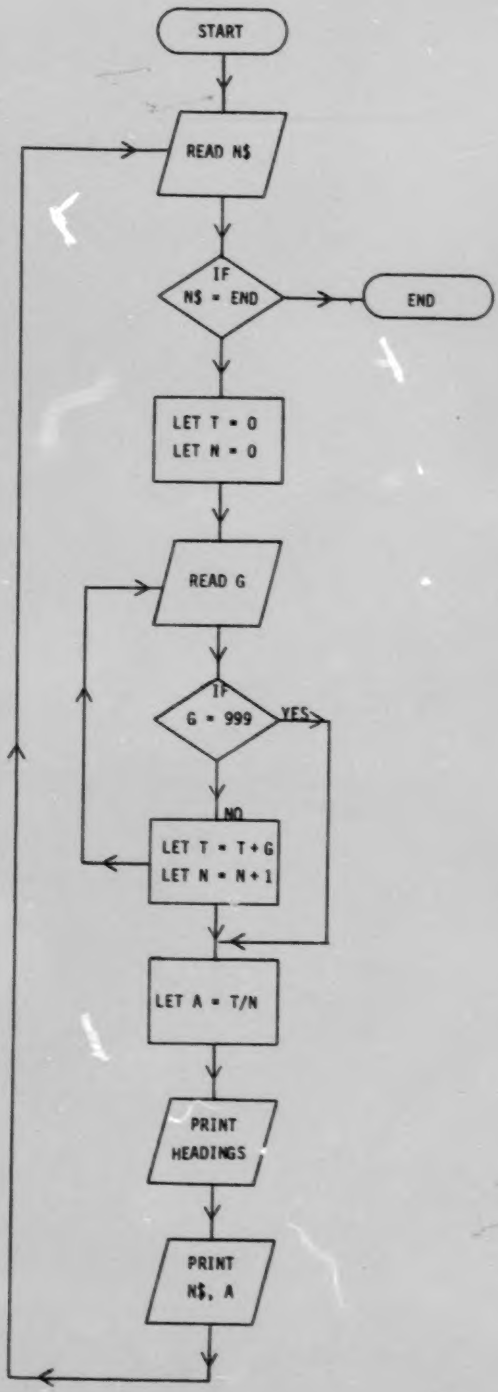
c. Study

d. Put away material

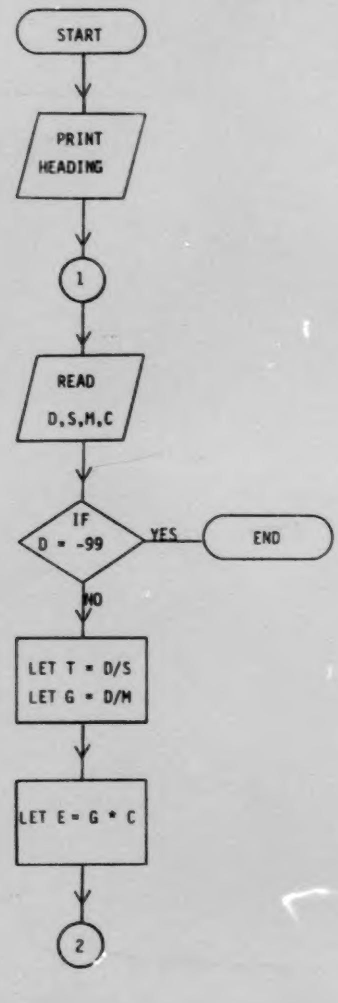
e. End

Assignment Sheet #3

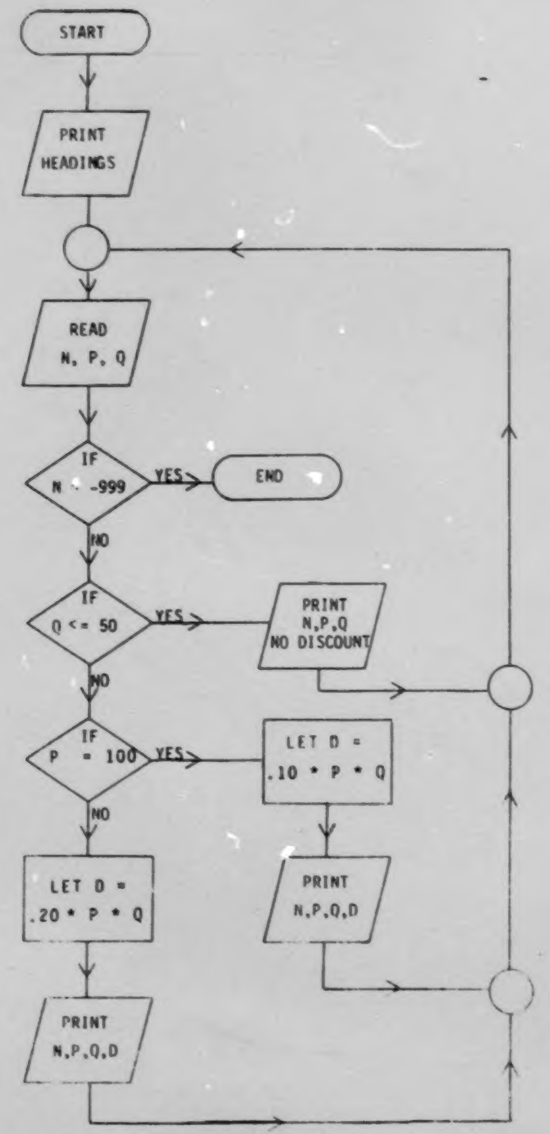
1. Program #1


START
READ N$
IF
N$ = END
END
LET T = 0
LET N = 0
READ G
IF
G = 999
YES
NO
LET T = T+G
LET N = N+1
LET A = T/N
PRINT
HEADINGS
PRINT
N$, A

2. Program #2

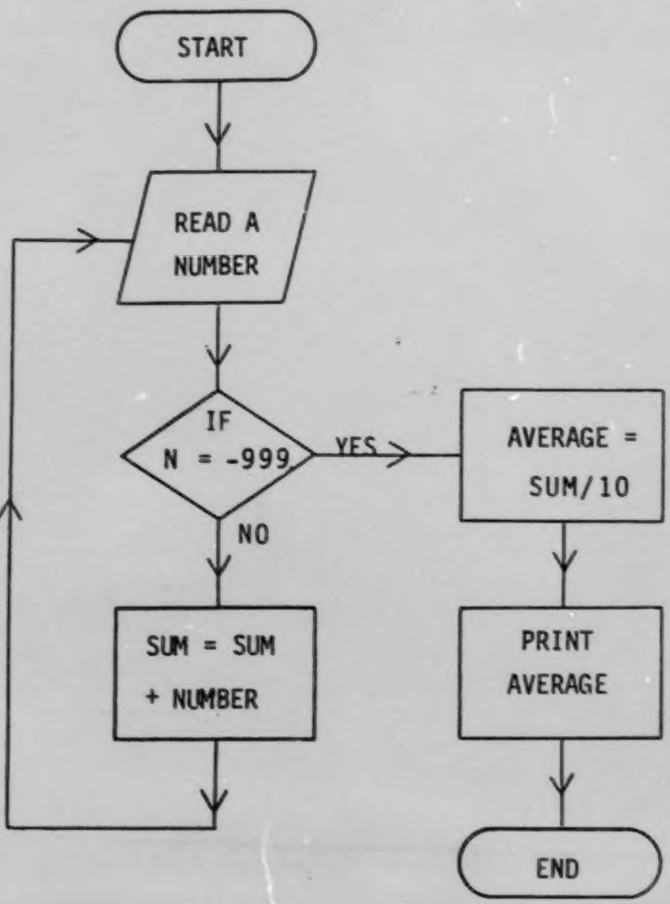

START
PRINT HEADING
1
READ D,S,M,C
IF D = -99
YES
END
NO
LET T = D/S
LET G = D/M
LET E = G * C
2
2
PRINT DISTANCE
PRINT AVERAGE SPEED
PRINT TIME OF TRIP
PRINT MILEAGE RATE
PRINT COST PER GALLON
PRINT ESTIMATED TRIP EXPENSE
1

3. Program #3


START
PRINT HEADINGS
READ N, P, Q
IF N = -999
YES
END
NO
IF Q <= 50
YES
PRINT N,P,Q NO DISCOUNT
NO
IF P = 100
YES
LET D = .10 * P * Q
PRINT N,P,Q,D
NO
LET D = .20 * P * Q
PRINT N,P,Q,D

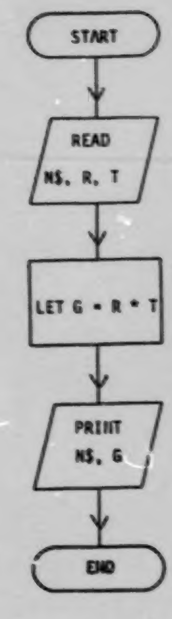
Assignment Sheet #4

```
100 REM  THIS PROGRAM READS 10 NUMBERS, FINDS THE SUM OF THOSE
110 REM  NUMBERS, AND THE AVERAGE OF THOSE NUMBERS
120 REM          N = NUMBER
130 REM          S = SUM
140 REM          A = AVERAGE
150 READ N
160 IF N = -999 THEN 190
170 LET S = S + N
180 GO TO 150
190 LET A = S/10
200 PRINT "THE AVERAGE IS ";A
300 DATA 10,34,23,18,56,29,88,64,42,72,-999
999 END
```

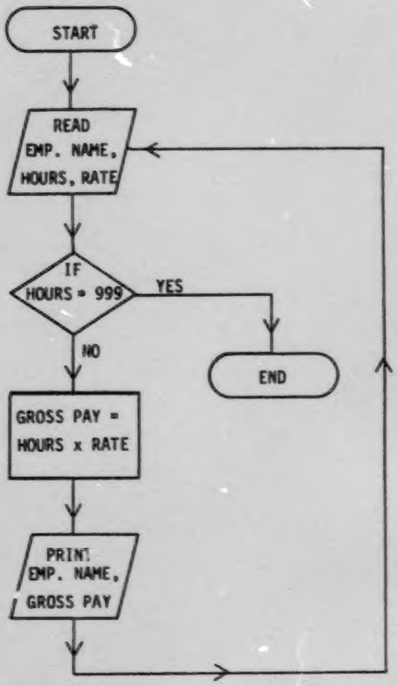

START
READ A NUMBER
IF N = -999
YES
NO
AVERAGE = SUM/10
SUM = SUM + NUMBER
PRINT AVERAGE
END

Assignment Sheet #5

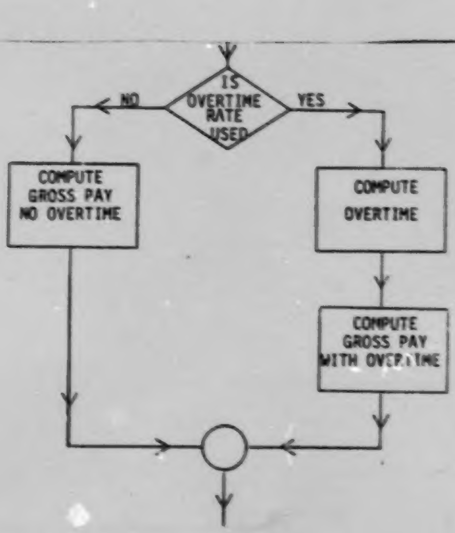
1.


START
READ
N$, R, T
LET G = R * T
PRINT
N$, G
END

2.


START
READ
EMP. NAME,
HOURS, RATE
IF
HOURS = 999
YES
END
NO
GROSS PAY =
HOURS x RATE
PRINT
EMP. NAME,
GROSS PAY

3.


IS
OVERTIME
RATE
USED
NO
YES
COMPUTE
GROSS PAY
NO OVERTIME
COMPUTE
OVERTIME
COMPUTE
GROSS PAY
WITH OVERTIME

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

NAME______________________

TEST

1. Match the terms on the right with the correct definitions:

______a. A programming aid designed to facilitate the coding of programs in a clear, concise manner with a minimum of ambiguity

______b. Error in the grammar or in the form of the programming language

______c. Step-by-step process described in terms of instructions a computer can follow

______d. Connects other symbols and indicates the sequence of operations

______e. Represents a decision, question, or comparison that determines which alternative paths are to be followed

______f. Graphic representation of order of operations in data processing system or program in which symbols are used to represent processing operations and data flow

______g. A supplemental means for storing data

______h. Connects separated portions of a flowchart without drawing lines between the parts

______i. Represents the beginning or ending of a program

______j. Plastic or metallic card containing both system and program flowcharting symbols as cutout forms

______k. Designates exit from or entry to another page

1. Auxiliary storage
2. Code
3. Coding forms
4. Connector symbol
5. Debug
6. Decision symbol
7. Documentation
8. Flow direction
9. Flowchart
10. Flowcharting template
11. Input/Output symbol
12. Logical errors
13. Offpage connector
14. Processing symbol
15. Program
16. Syntax error
17. Terminal symbol
18. Testing

______l. Errors which occur when the programmer does not thoroughly understand a phase of the problem, fails to account for all situations that may arise during processing, or misinterprets some steps when coding

______m. Represents an input or output operation

______n. The illustrations and explanations that should accompany the computer printout of a "working" program that include both general information needed by the user of the program and the specific information needed by the computer operator or by the computer programmer if the program later needs updating

______o. Represents calculations, initializations, and other processing functions

______p. To translate and write information in a programming language

______q. Task of detecting, locating, and eliminating mistakes or errors in a program

______r. Run program against representative data to ensure an adequacy of the program or system for regular use

2. Select true statements concerning the program development cycle by placing an "X" in the appropriate blanks.

______a. The steps in the development of a program include:

(1) define the problem
(2) plan the solution
(3) code the solution
(4) review the program and debug
(5) enter program into the computer
(6) test and debug
(7) document

______b. The program flowchart is drawn before coding a program

______c. The drawing of the flowchart is part of the planning step of the development of a program

______d. Only the programmer reviews the program ______e. Only valid data is needed in the testing state of the cycle

______f. A final flowchart is included in documentation

______g. The clear documentation of the program is a mark of a well written program

______h. Variable names and data used are not included in documentation

______i. A programmer enters the program before it is coded on a coding form

______j. The testing of the program verifies that the program will consistently produce correct results

3. Match the following basic program flowcharting descriptions with their symbols

______a. Flow Direction

______b. Processing symbol

______c. Terminal symbol

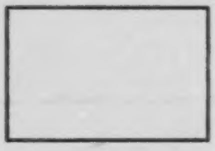
______d. Decision symbol

______e. Input/output symbol

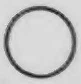
______f. Connector symbol

______g. Offpage connector

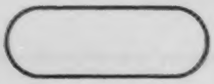
1.



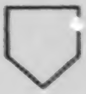
2.



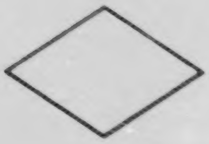
3.



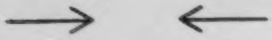
4.



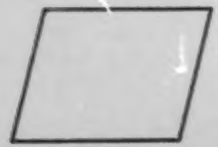
5.



6.



7.

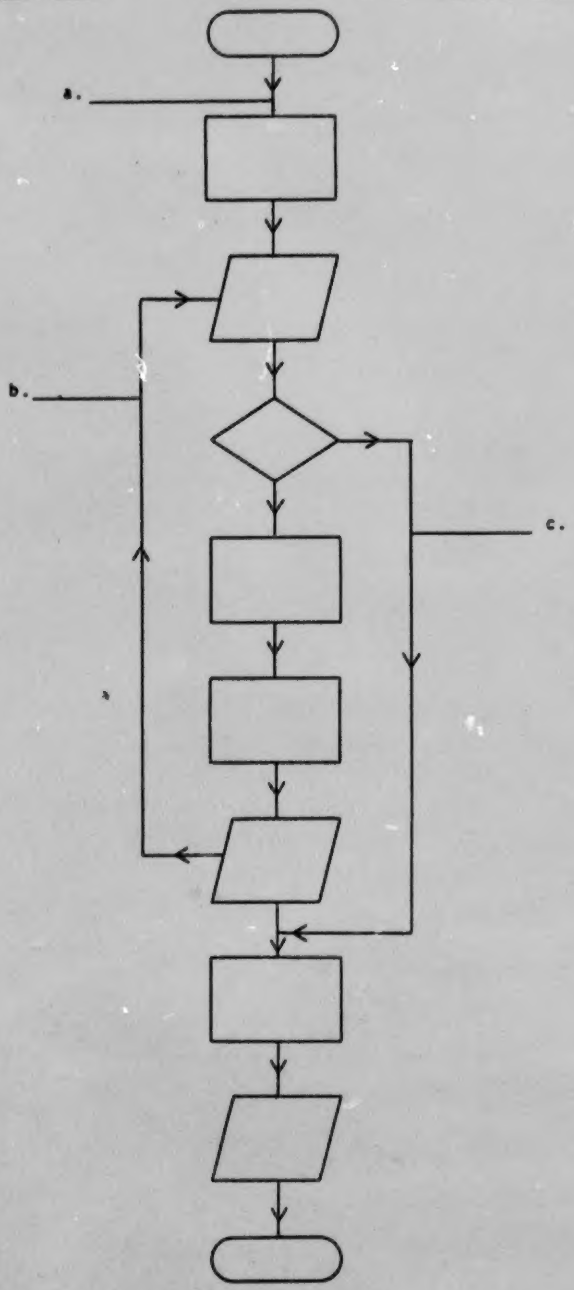


4. Match the sections of the structured flowchart below to the appropriate design by placing the corresponding letter in the blank provided.

   S. Sequence structure

   L. Loop structure

   C. Conditional structure



5. Code, enter, and RUN programs using steps outlined in flowcharts.

6. Complete a program flowchart.

7. Construct program flowcharts from coded programs.

8. Demonstrate the ability to perform the steps in the program development cycle.

9. Match structured flowcharts to their correct basic design.

(NOTE: The above activities 5 - 8 should be completed before the test is taken. If these activities have not been accomplished prior to the test, ask your instructor when they should be completed.)

PROGRAM DEVELOPMENT CYCLE
AND FLOWCHARTING
UNIT VI

ANSWERS TO TEST

1.
   - a. 3
   - b. 16
   - c. 15
   - d. 8
   - e. 6
   - f. 9
   - g. 1
   - h. 4
   - i. 17
   - j. 10
   - k. 13
   - l. 12
   - m. 11
   - n. 7
   - o. 14
   - p. 2
   - q. 5
   - r. 18

2.
   - a.
   - b.
   - c.
   - f.
   - g.
   - j.

3.
   - a. 6
   - b. 1
   - c. 3
   - d. 5
   - e. 7
   - f. 2
   - g. 4

4.
   - a. S
   - b. L
   - c. C

5. Evaluated to the satisfaction of the instructor.

6. Evaluated to the satisfaction of the instructor.

7. Evaluated to the satisfaction of the instructor.

8. Evaluated to the satisfaction of the instructor.

9. Evaluated to the satisfaction of the instructor.

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

UNIT OBJECTIVE

After completion of this unit, the student should be able to match terms associated with PRINT USING statements and TAB functions, select true statements concerning PRINT USING statements, determine the output for numeric and string formatted fields, and select true statements concerning TAB functions. The student should also be able to code, enter, and RUN programs that use different numeric and string format fields and values to produce formatted output; code, enter, and RUN PRINT USING statements for selected output; and flowchart, code, enter, and RUN programs using PRINT USING statements and TAB functions. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with PRINT USING statements and TAB functions.
2. Select true statements concerning the PRINT USING statements.
3. Determine the output for numeric and string formatted fields.
4. Select true statements concerning the TAB function.
5. Enter, and RUN a program that uses different numeric format fields and values to produce formatted output.
6. Enter, and RUN a program that uses different string format fields and values to produce formatted output.
7. Enter, and RUN a program that uses three different string format fields and values to produce formatted output.
8. Code, enter, and RUN programs using PRINT USING statements to produce formatted output for selected output.
9. Flowchart, code, enter, and RUN programs using PRINT USING statements and TAB functions.

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

  A. Objective sheet

  B. Information sheets

  C. Assignment sheets

    1. Assignment Sheet #1--Enter, and RUN a program that uses different numeric format fields and values to produce formatted output

    2. Assignment Sheet #2--Enter, and RUN a program that uses different string format fields and values to produce formatted output

    3. Assignment Sheet #3--Enter, and RUN a program that uses three different string format fields and values to produce formatted output

    4. Assignment Sheet #4--Code, enter, and RUN programs using PRINT USING statements to produce formatted output for selected output

    5. Assignment Sheet #5--Flowchart, code, enter, and RUN programs using PRINT USING statements and TAB functions

  D. Answers to assignment sheets

  E. Test to test

  F. Answers

II. References:

Refer to Unit I references.

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

INFORMATION SHEET

I. Terms and definitions

A. PRINT USING statement - a special form of the PRINT statement that includes special features to print both numeric and string data

B. Format field - a statement that describes the arrangement used for a PRINT USING statement

C. Field - in a PRINT USING statement, a set of character positions in which something is printed

D. List - in a PRINT USING statement, constants and variables to be printed according to the format defined in the format field

E. TAB function - a function which is used with the PRINT statement to tabulate to specific column positions

F. Argument - a valid BASIC arithmetic expression enclosed in parentheses that is used with BASIC functions

G. Truncated - digits are dropped in contrast to rounding of numbers

H. Function - subroutines stored in the library of the computer's memory that are used in programs when special operations are to be performed

II. PRINT USING statement

(NOTE: The PRINT USING statement, sometimes called image statements is not compatible with all computers. Various types of PRINT USING statements are used with different computers. The manufacturer's manual should be consulted for the types of statements to be used on each particular computer.)

A. Purpose

1. Gives control over the appearance and location of printed data

2. Useful for producing tables and forms

INFORMATION SHEET

3. Used for printed output of both numeric and string values
4. Can be used to round numeric data
5. Aligns decimal points and justifies numbers on the right
6. Prints and adjusts column headings more easily

B. General format

1. Line number PRINT USING <format field>, <list> (PDP 11)

   or

   Line number PRINT USING <format field>; <list> (TRS-80)

   a. Format field can be:
      1) Numeric format
      2) String format
      3) Constants
      4) One or more fields of numeric format, string format, and/or constants

   b. List can be:
      1) Constant
      2) Numeric variable
      3) String variable
      4) One or more constants and/or variables

      (NOTE: A comma will be used in all examples; TRS-80 users will have to use a semicolon in place of the comma.)

2.

```
100 PRINT USING "###  ###",A,1          (PDP 11)
```

or

```
100 PRINT USING "###  ###";A,1          (TRS-80)
```

INFORMATION SHEET

a. Statement inside quotes shows format of field to be printed

b. Variables and constants (A and 1 in example) show values to be printed according to the format defined in the quotes

Example:

```
10 LET A = 13
20 PRINT USING "### ###";A,1
99 END

RUN
 13   1
```

C. Numeric data

1. The # symbol indicates the number of digits to be reserved for each number to be printed

2. One # symbol is used for each digit to be printed

Example:

```
10 PRINT USING "#### ###",1234,123

RUN

1234 123
```

or

```
10 LET A = 1111
20 LET B = 444
30 PRINT USING "#### ###",A,B

RUN

1111 444
```

3. A . (decimal point) followed by # symbols indicates the number of decimal places to be printed

INFORMATION SHEET

Example:

```
10 PRINT USING "##.# ###.##",24.2,24.12,24.23,124.14

RUN

24.2  24.12
24.2 124.14
```

4. Too many digits to the left of the decimal point cause a % to be printed in front of the data to indicate field format size error exists in the PRINT USING statement

   Example:

   ```
   10 PRINT USING "##",123

   RUN

   %123
   ```

5. Too many digits in the number to the right of the decimal point cause the number to be rounded off

   a. If the number to the right of the last decimal space formatted is equal to or greater than five, the decimal value will increase by 1

   ```
   100 PRINT USING "###.##",123.567

   RUN

   123.57
   ```

   b. If the number to the right of the last decimal space formatted is less than 5, the decimal value remains the same

   ```
   100 PRINT USING "###.##",123.444

   RUN

   123.44
   ```

6. Zeros are added to fill the spaces when fewer digits than # symbols are used to the right of the decimal point

INFORMATION SHEET

```
100 PRINT USING "###.##",200

RUN

200.00
```

7. Blank spaces are added to fill the spaces when fewer digits than # symbols are to the left of the decimal point

```
100 PRINT USING "####.##", 3.45

RUN

  3.45
```

8. One # symbol is required for the - sign in negative numbers

```
100 PRINT USING "### ###",-23, -123

RUN

-23 % -123
```

9. A , (comma) can be printed in long numbers for easier readability

   a. Placement of the comma is determined by the placement between the # symbols

   b. The comma is only printed when it is required

```
100 READ A
110 IF A = -999 THEN 999
200 PRINT USING "###,###.##",A
210 GO TO 100
300 DATA 100,1000,10000,100000,-1000,-999
999 END

RUN

    100.00
  1,000.00
 10,000.00
100,000.00
 -1,000.00
```

INFORMATION SHEET

D. String fields

1. A backslash (\) is used on some computers (PDP-11); a percent sign (%) is used on other computers (TRS-80) to indicate string format

   a. Length of string field is determined by a pair of the backslashes (\\) or percent signs (%%) and the spaces between

   b. The length of the field will be two plus the number of spaces between the backslashes or percent signs

   Example:

   For the PDP 11

```
100 PRINT USING "\     \",JOHNSON

JOHNSON
```

   For the TRS-80

```
100 PRINT USING "%     %";JOHNSON

JOHNSON
```

   c. If enough space is not allowed for the string data, the characters to the right of the formatted output field will not be printed

```
100 PRINT USING "%  %",JOHNSON

RUN

JOHN
```

2. A ! (exclamation point) causes the computer to use the first string character of the current value to be printed

```
100 PRINT USING "!!!","PARKVIEW", "HIGH",
"SCHOOL"

RUN

PHS
```

INFORMATION SHEET

```
100 PRINT USING "! ! !","PARKVIEW","HIGH",
"SCHOOL"

RUN

P H S

10 LET A$ = "CAROL"
20 LET M$ = "ANN"
30 LET L$ = "SMITH"
40 PRINT USING "!. !. %   %",A$,M$,L$
99 END

RUN

C. A. SMITH
```

E. Characters included in the quotation marks in the format field will be printed as a string literal

```
100 LET A = 10
110 LET A$ = "WEBSTER"
120 PRINT USING "THE NUMBER IS ###",A
140 PRINT USING "THE INSTRUCTOR'S NAME IS %      %",A$
999 END

RUN

THE NUMBER IS  10
THE INSTRUCTOR'S NAME IS WEBSTER
```

F. Special formatting symbols

1. $$

   a. Prints $ before each field

   b. Causes $ to float

   c. First $ causes the $ to be printed

   d. Second $ is used as if it were another # symbol

   e. Useful for writing checks

INFORMATION SHEET

Example:

```
100 READ A
110 IF A = -999 THEN 999
120 PRINT USING "$$##.##",A
130 GO TO 100
140 DATA 344.44,34.34,2,1.44,-999
999 END
```

```
RUN

$344.44
 $34.44
  $2.00
  $1.44
```

2. $

   a. A single $ printed causes a $ to be printed as the left-most value of numeric data

   b. Does not float

   Example:

```
100 READ A
110 IF A = -999 THEN 999
120 PRINT USING "$###.##",A
130 GO TO 100
140 DATA 344.44,34.34,3,1.44,-999
999 END
```

```
RUN

$344.44
$ 34.34
$  3.00
$  1.44
```

3. **

   a. Double asterisks cause the * to float

   b. Fills number fields with asterisks when the number contains fewer digits than the width of the field

   c. No asterisks are printed if the number is the width of the field

INFORMATION SHEET

Example:

```
10 READ A
20 IF A = -999 THEN 999
30 PRINT USING"**###.##",A
40 GO TO 10
50 DATA 1,3456,4,1,44,-999
99 END
```

RUN

```
***1.00
3456.00
***4.00
***1.44
```

d. Useful when writing checks

4. *

   a. A single * (asterisk) in the format field causes a single * to be printed

   b. Does not float

Example:

```
10 READ A
20 IF A = -999 THEN 99
30 PRINT USING "*####.##",A
40 GO TO 10
50 DATA 4000.33,44,34.44,1.45,-999
99 END
```

RUN

```
*4000.33
44444.44
*  34.44
*   1.45
```

5. -

   a. A - sign can be included in field format after a numeric value

   b. The - sign will print after the number if the number is negative

   c. A blank space is printed if the number is positive

INFORMATION SHEET

Example:

```
10 READ A,B,C
20 IF A = -999 THEN 99
30 PRINT USING "###-   ###-  ###-"A,B,C
40 DATA -100,-30,200,100,30,-200,-999
99 END

RUN

100-    -30   200
-100     30   200-
```

G. Assignment of format field using a LET statement

1. Formats can be assigned to string variable names through the LET statement
2. This must be done before the PRINT USING statement is executed
3. Can be used as many times as needed
4. More than one format field can be assigned to variables

Example:

```
10 LET F$ = "####"
20 LET F1$ = "#,###"
30 LET F2$ = "$#,###"
40 LET F3$ = "NAME IS %       %"
50 LET F4$ = "TOTAL IS $$###,###.##"
60 LET A = 1000
70 LET A$ = KENNETH
80 PRINT USING F$,A
90 PRINT USING F1$,A
100 PRINT USING F2$,A
110 PRINT USING F3$,A$
120 PRINT USING F4$,A
999 END

RUN

1000
1,000
$1,000
NAME IS KENNETH
TOTAL IS   $1,000.00
```

INFORMATION SHEET

H. Line formatting

1. When more variables or constants than pound signs are in the format field, the values will be printed in one of two ways:

   a. The values will be continued on the next line using the same format

   Example:

```
10 LET A = 1
20 LET B = 2
30 LET C = 3
40 PRINT USING "#",A,B,C
99 END

RUN

1
2
3
```

```
10 LET A = 1
20 LET B = 2
30 LET C = 3
40 LET D = 4
50 LET E = 5
60 LET F = 6
70 PRINT USING "# # #",A,B,C,D,E,F
99 END

RUN

1 2 3
4 5 6
```

   (NOTE: The PDP 11/34 follows the above rule.)

   b. The values will continue to print on the same line

   Example:

```
10 LET A = 1
20 LET B = 2
30 LET C = 3
40 PRINT USING "#";A,B,C
99 END
```

INFORMATION SHEET

```
RUN

123

10 LET A = 1
20 LET B = 2
30 LET C = 3
40 LET D = 4
50 LET E = 5
60 LET F = 6
70 PRINT USING "# # # ";A,B,C,D,E,F
99 END

RUN

1 2 3 4 5 6
```

2. If fewer constants or variables than pound signs are in the format field, the extra format fields will be filled with blank spaces

Example:

```
10 PRINT USING "### ### ### ###",25
99 END

RUN

 25
```

III. TAB function

A. Used to print a large number of spaces

B. Especially useful in printing headings for reports and tables

C. General format

1.

```
                         100 PRINT TAB(20);"ANSWER IS",A
                          |    |    |   | |            |
a.  Line Number___________|    |    |   | |            |
                               |    |   | |            |
b.  The word PRINT_____________|    |   | |            |
                                    |   | |            |
c.  The word TAB____________________|   | |            |
                                        | |            |
d.  Parentheses with column             | |            |
    number (argument)___________________| |            |
                                          |            |
e.  A semicolon___________________________|            |
                                                       |
f.  The variable and/or constant to be printed_________|
```

INFORMATION SHEET

2. Argument (column number) may be:

   a. Constant

   ```
   10 PRINT TAB(10)
   ```

   b. Variable

   ```
   10 PRINT TAB(X)
   ```

   c. Expression

   ```
   10 PRINT TAB(A + B)
   ```

D. Operation of TAB function

1. Operates similar to a TAB key on a typewriter

2. Computer inserts the number of spaces enclosed in parentheses from the left margin

3. A semicolon after the TAB( ) will cause the printing to start one space to the right of the value specified in parentheses

   Example:

   ```
   10 PRINT TAB(0);-1;TAB(5)"APPLE"
   ```

   ```
   -1   APPLE
   ```

4. A comma after the TAB function will cause the printing to start at the first field to the right of the value specified in parentheses

   Example:

   ```
   10 PRINT TAB(3),"TODAY IS"
   ```

   ```
      TODAY IS
   ```

5. When the value of the column number in the parentheses is less than or equal to the current position of the print line, the TAB is ignored

   Example:

   ```
   10 PRINT "123";TAB(2)"456"
   ```

   ```
   123456
   ```

INFORMATION SHEET

(NOTE: On some computers, the above will cause the computer to advance to the next line to the position indicated.)

E. Rules

1. Column number (argument) must be a positive value

2. If argument is a decimal number, the number will be truncated

   (Note: The value after the decimal point is ignored when a number is truncated.)

3. More than one TAB is allowed in a PRINT line

4. Arguments (column numbers) must be progressively larger

```
10 PRINT TAB(10);N;TAB(20);B;TAB(30);C
```

5. Arguments need to be less than the number of spaces allowed per line of print on the terminal (usually 72)

6. Room must be allowed between TABS for data to be printed; TABS will be ignored if there is not enough room for the data to be printed

7. TAB functions cannot be included in a PRINT USING statement

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

ASSIGNMENT SHEET #1--Enter and RUN a program to demonstrate the output produced by numeric formatted fields

Directions: Enter the following program into the computer using the name PROG1. SAVE the program.

```
10 REM   THE FOLLOWING PROGRAM IS USED TO DEMONSTRATE THE
20 REM   OUTPUT PRODUCED BY THE PRINT USING STATEMENT
30 REM   THE USER WILL INPUT A FIELD FORMAT AND A NUMBER WHEN
40 REM   THE QUESTION IS ASKED, THE COMPUTER WILL THEN SHOW
50 REM       THE OUTPUT PRODUCED BY THAT FORMAT
60 REM
70 REM          Q$ = YES OR NO FOR CONTINUING
80 REM          N$ = FORMAT FIELD
90 REM          N  = NUMERIC DATA TO BE FORMATTED
99 PRINT:PRINT
100 INPUT "DO YOU WISH TO CONTINUE - ANSWER Y FOR YES OR N FOR NO";Q$
110 IF Q$ = "N" THEN 999
114 PRINT:PRINT
115 PRINT:PRINT
116 PRINT "THE FIELD FORMAT MUST BE ENCLOSED IN QUOTATION MARKS"
119 PRINT:PRINT
120 INPUT "WHEN THE ? APPEARS, TYPE THE NUMERIC FIELD FORMAT";N$
125 PRINT:PRINT
130 INPUT "WHEN THE ? APPEARS, TYPE THE NUMERIC VALUE";N
140 PRINT USING N$,N  (On TRS-80 use 140 PRINT USING N$;N)
150 GO TO 100
999 END
```

RUN the program using the following numeric formatting fields:

	FORMAT FIELD	VALUE
1.	##.#	14.444
2.	##.#	14.445
3.	##.#	144.455
4.	$###.##	245.55
5.	$###.##	24.4
6.	$###.##	4566.45
7.	$$###	4566.78
8.	$$###	456.44
9.	###,###.##	45677.77
10.	###,###.##	45.55
11.	$###,###.##	345.55
12.	$###,###.##	334444.00
13.	$###,###.##	34432.22
14.	###	-30

ASSIGNMENT SHEET #1

15.	###	-300
16.	**#####.##	134.00
17.	**#####.##	1234567.34
18.	**#####.##	123456.789
19.	*#####.##	99999.99
20.	*#####.##	9.999
21.	###.##-	4.44
22.	###.##-	-4.44
23.	THE ANSWER IS ####.##	435.66
24.	THE TOTAL IS $$,###.##	4567.77

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

ASSIGNMENT SHEET #2--Enter and RUN a program to demonstrate the output produced by string formatted fields

Directions: Enter the following program into the computer using the name PROG2. SAVE the program.

```
10 REM  THE FOLLOWING PROGRAM IS USED TO DEMONSTRATE THE
20 REM  OUTPUT PRODUCED BY THE PRINT USING STATEMENT.
30 REM  THE USER WILL INPUT A STRING FIELD FORMAT AND STRING
40 REM  DATA WHEN THE QUESTION APPEARS ON THE SCREEN.
50 REM  THE COMPUTER WILL THEN SHOW THE OUTPUT PRODUCED BY THE
60 REM  FORMAT SPECIFIED.
60 REM
70 REM          Q$ =   YES OR NO FOR CONTINUING
80 REM          N$ =   FORMAT FIELD
90 REM         N1$ =   STRING DATA TO BE FORMATTED
99 PRINT:PRINT
100 INPUT "DO YOU WISH TO CONTINUE - ANSWER Y FOR YES OR N FOR
NO";Q$
110 IF Q$ = "N" THEN 999
114 PRINT:PRINT
115 PRINT:PRINT
116 PRINT "THE FIELD FORMAT MUST BE ENCLOSED IN QUOTATION
MARKS"
119 PRINT:PRINT
120 INPUT "WHEN THE ? APPEARS, TYPE A STRING FIELD FORMAT";N$
125 PRINT:PRINT
130 INPUT "WHEN THE ? APPEARS, TYPE THE STRING VALUE";N1$
140 PRINT USING N$,N1$
150 GO TO 100
999 END
```

RUN the program using the following string formatting fields and data:

(NOTE: Use % sign instead of \ sign for the TRS-80.)

	FORMAT FIELD	VALUE
1.	`!`	CATHRYN
2.	`!`	ARKANSAS
3.	`\          \`	TENNESSEE
4.	`\   \`	TENNESSEE
5.	`\\`	CALIFORNIA
6.	`\         \`	THOMAS
7.	`\         \`	ALAN
8.	`\         \`	JAY

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

ASSIGNMENT SHEET #3--Enter and RUN a program to demonstrate the output produced by three string formatted fields

Directions: Enter the following program into the computer using the name PROG3. SAVE the program.

This program will require three formatted string fields and three string values.

```
10 REM  THE FOLLOWING PROGRAM IS USED TO DEMONSTRATE THE
20 REM  OUTPUT PRODUCED BY THE PRINT USING STATEMENT.
30 REM  THE USER WILL INPUT A STRING FIELD FORMAT FOR THREE
35 REM  STRING VALUES.
40 REM  THE STRING DATA WILL THEN BE ENTERED WHEN THE QUESTION
50 REM  MARK APPEARS ON THE SCREEN.
60 REM  THE COMPUTER WILL THEN SHOW THE OUTPUT PRODUCED BY
65 REM  EACH FORMAT.
70 REM          Q$ =   YES OR NO FOR CONTINUING
80 REM          N$ =   FORMAT FIELD
90 REM         N1$ =   STRING DATA FOR FIRST VALUE
91 REM         N2$ =   STRING DATA FOR SECOND VALUE
92 REM         N3$ =   STRING DATA FOR THIRD VALUE
99 PRINT:PRINT
100 INPUT "DO YOU WISH TO CONTINUE - ANSWER Y FOR YES OR N FOR
NO";Q$
110 IF Q$ = "N" THEN 999
114 PRINT:PRINT
115 PRINT:PRINT
116 PRINT "THE FIELD FORMAT MUST BE ENCLOSED IN QUOTATION
MARKS"
119 PRINT:PRINT
120 INPUT "WHEN THE ? APPEARS, TYPE THE STRING FIELD
FORMAT";N$
125 PRINT:PRINT
130 INPUT "WHEN THE ? APPEARS, TYPE THE STRING VALUE OR
VALUES";N1$,N2$,N3$
140 PRINT USING N$,N1$,N2$,N3$
150 GO TO 100
999 END
```

ASSIGNMENT SHEET #3

RUN the program using the following string formatting fields:

	FORMAT FIELD	VALUE
1.	`!!!`	Ben, Alfred, McGuire
2.	`! ! !`	Ben, Alfred, McGuire
3.	`!. !. !.`	Ben, Alfred, McGuire
4.	`!! \         \`	Sandra Ann Williams
5.	`! ! \          \`	Sandra Ann Williams
6.	`!. !. \          \`	Sandra Ann williams
7.	`\         \ \\ \          \`	Kenneth Knox McConnell
8.	`\\ \         \ \          \`	Kenneth Knox McConnell
9.	`LAST NAME IS \         \`	Smith
10.	`LAST NAME IS \         \`	Harrison

(NOTE: Use % sign in place of the \ sign on TRS-80.)

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

ASSIGNMENT SHEET #4--Code, enter, and RUN programs using PRINT USING statements to produce formatted output for selected output

Directions: Use PRINT USING statements to produce the following formatted output. Use READ/DATA statements to assign values to variables.

1. The following output is to be printed for variables A, B, and C:

```
 100     24     300
  50    100    1000
2000   6000       3
```

2. The following output is to be printed for variables A, B, and C:

```
   10.00          .24     -1000.00
-2000.00         2.67         3.96
     .40     32455.37        42.62
```

3. The following output is to be printed for variables A, B, and C:

```
$ 246.52          $    4.07          $ 100.23
$   3.76          $   24.07          $2476.00
$  24.44          $     .01          $  42.33
```

4. The following output is to be printed for variables A, B, and C:

```
   $2.45          $345.55           $23.00
$1455.55       $123455.55        $12333.33
  $34.44        $12345.55            $2.22
```

5. The following output is to be printed for variables A, B, and C:

```
% 12345         3.56          110.99
   4444        23.44       123456.55
```

6. The following output is to be printed for variable X:

```
TOTAL EARNINGS $44,555.55
```

ASSIGNMENT SHEET #4

7. The following output is to be printed for variable C:

```
1111.90
*111.90
**11.90
***1.90
****.90
```

8. The following output is to be printed for variable C:

```
$1111.98
$$111.98
$$$11.98
$$$$1.98
```

9. Using data of 246.367, 299.626, 255.9876, print one vertical column of numbers with numbers rounded to two decimal places.

10. Using data of 24,444.90, 3,234.90, 54,333.33, 2,000, 42.50, 2,588.90, 43,555, 200, -200, and .02, print output with commas when appropriate, 5 rows of output with 2 columns of numbers in each row. Leave 4 spaces between columns. Round the numbers to 1 decimal place.

11. Using the data of Jimmy Carter, Gerald Ford, Richard Nixon, and Lyndon Johnson, print the names in 2 rows with 2 names per row. Leave 10 spaces between the names.

12. Print the names listed in number 11 with the first initials and last name.

13. Print the first character of your first and middle name with periods following each and your last name in full.

14. Print your room number and instructor's name in the following format:

```
DATA PROCESSING IS IN ROOM 3209

THE INSTRUCTOR IS THOMAS EDISON
```

15. Using the following data 2.96,2.99,100,2345.93,3.42,10, print the data with floating asterisks in 3 rows with 2 columns of numbers in each row.

16. Using the data in problem 15, print the data with asterisks which do not float.

17. Using the data 24.26,29.26,123.25,-24.26,-29.26,-123.25, print the data with trailing + or - signs.

ASSIGNMENT SHEET #4

18. Using the data in problem 17, print the data with trailing - signs only.

19. Assign the format field of "### ### ###.##" to F$. Print the values of the following variables using the format assigned to F$:

A = 200
B = 300
C = 10.22

20. Assign the format field of "\   \ \   \ IS ## YEARS OLD" to F1$. Print the values of the variables N$,N1$,A using the format of F1$ for the following values:

ANNA TAYLOR 45
DONALD BINZ 25
ALBERT DAVIS 3

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

ASSIGNMENT SHEET #5--Flowchart, code, enter, and RUN programs using PRINT USING statements and TAB functions

1. Flowchart, code, enter, and RUN a program to ask for a person's first name, middle name, last name, and the amount of money a check is to be written for (INPUT statements). Print out the person's name in the following format:

   ```
   PAY TO THE ORDER OF

   C. R. MARTIN

   *********456.98
   ```

   Use a LET statement to assign the PRINT USING format for both the printing of the name and amount of the check.

   Use your name for the test data and the amount of 3,444.44 for the check.

2. Flowchart, code, enter, and RUN a program to do the following:

   Enter the following data for a phone bill for one year:

January	32.56
February	34.66
March	55.55
April	323.33
May	32.56
June	32.56
July	55.44
August	36.99
September	65.55
October	56.44
November	32.56
December	66.66

   Print out the total for each month and the total for each half year.

   Compute both the average monthly bill and the average daily bill based on 365 days.

   Print the title YEARLY TELEPHONE BILL using the TAB function.

   Use the PRINT USING statement to print the output in an attractive form.

ASSIGNMENT SHEET #5

3. Flowchart, code, enter, and RUN a program for the following problem. Use PRINT USING statements for the formatted output. Use the TAB function to print the heading for the output.

   The ANDERSON APPLIANCE COMPANY offers different discounts to different customers. These discounts are entered as codes. Code 1 is for a 5% discount. Code 2 is for a 10% discount. Code 3 is for a 15% discount. Code 4 is for a 20% discount.

   For the following sales, compute the discount and net amount of the order. Use the following data:

CUSTOMER'S NAME	CODE NUMBER	AMOUNT OF ORDER
JANE ANDERSON	4	$100.00
ROBERT BALL	1	150.00
ANDREW CARMON	2	300.00
SUSAN DREW	3	1,000.00
TOM ELF	1	579.00
BRENDA FELTS	2	199.98
MIKE GATES	2	235.66
ANNE HIGGONS	4	2,456.96
BRENT ISOM	4	3,456.78

   Print the customer's name, code number, amount of order, amount of discount, and net amount with appropriate headings.

   Calculate the total amount ordered, total amount of discount, and total net amount. Print the word TOTAL and the amounts at the bottom of the above information.

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

Evaluated to the satisfaction of the instructor.

Assignment Sheet #2

Evaluated to the satisfaction of the instructor.

Assignment Sheet #3

Evaluated to the satisfaction of the instructor.

Assignment Sheet #4

1.

```
100 READ A,B,C
105 IF A = -999 THEN 999
110 PRINT USING "####    ####    ####",A,B,C
120 GOTO 100
800 DATA 100,24,300,50,100,1000,2000,6000,3
810 DATA -999,0,0
999 END
```

2.

```
100 READ A,B,C
105 IF A = -999 THEN 999
110 PRINT USING "#####.##    #$###.##    #####.##",A,B,C
120 GOTO 100
800 DATA 10,.24,-1000,-2000,2.67,3.96,.4,32455.37,42.62
810 DATA -999,0,0
999 END
```

3.

```
100 READ A,B,C
105 IF A = -999 THEN 999
110 PRINT USING "$####.##         $#####.##         $####.##"A,B,C
120 GOTO 100
800 DATA 246.52,4.07,100.23,3.76,24.07,2476,24.44,.01,42.33
810 DATA -999,0,0
999 END
```

4.
```
100 READ A,B,C
105 IF A = -999 THEN 999
110 PRINT USING"$$###.##        $$#####.##         $$####.##",A,B,C
120 GOTO 100
800 DATA 2.45,345.55,23,1455.55,123455.55,12333.33,34.44,12345.55
810 DATA 2.22,-999,0,0
999 END
```

5.
```
100 READ A,B,C
105 IF A = -999 THEN 999
110 PRINT USING "####        ##.##        ######.##",A,B,C
120 GOTO 100
800 DATA 12345,3.56,110.99,4444,23.44,123456.55
810 DATA -999,0,0
999 END
```

6.
```
100 X = 44555.55
105 PRINT
110 PRINT USING "TOTAL EARNINGS $##,###.##",X
999 END
```

7.
```
100 READ C
105 IF C = -999 THEN 999
110 PRINT USING "**##.##",C
120 GOTO 100
800 DATA 1111.90,111.90,11.90,1.90,.90
810 DATA -999
999 END
```

8.
```
100 READ C
105 IF C = -999 THEN 999
110 PRINT USING "$$###.##",C
120 GOTO 100
800 DATA 1111.98,111.98,11.98,1.98
810 DATA -999
999 END
```

9.
```
100 READ C
105 IF C = -999 THEN 999
110 PRINT USING "###.##",C
120 GOTO 100
800 DATA 246.367,299.626,255.9876
810 DATA -999
999 END
```

```
10. 100 READ C,D
    105 IF C = -999 THEN 999
    110 PRINT USING "##,###.#    ##,###.#",C,D
    120 GOTO 100
    800 DATA 24444.90,3234.90,54333.33,2000
    805 DATA 42.50,2588.90,43555,200,-200,.02
    810 DATA -999,0
    999 END

11. 100 READ A$,B$
    105 IF A$ = "LASTNAME" THEN 999
    110 PRINT USING "\             \          \             \",A$,B$
    120 GOTO 100
    800 DATA "JIMMY CARTER","GERALD FORD", "RICHARD NIXON"
    805 DATA "LYNDON JOHNSON"
    810 DATA "LASTNAME", "L"
    999 END

12. 100 READ A$,B$
    105 IF A$ = "LASTNAME" THEN 999
    110 PRINT USING "! \        \          ! \       \",A$,B$
    120 GOTO 100
    800 DATA JIMMY,CARTER,GERALD,FORD,RICHARD,NIXON,LYNDON,JOHNSON
    810 DATA "LASTNAME", "L"
    999 END

13. 100 READ A$,B$,C$
    110 PRINT USING "!. !. \              \",A$,B$,C$
    800 DATA ALICE,HAZEL,THOMPSON
    999 END

14. 100 READ A,B$
    110 PRINT USING "DATA PROCESSING IS IN ROOM ####",A
    120 PRINT USING "THE INSTRUCTOR IS \                \",B$
    800 DATA 3205, "EARL HILLARD"
    999 END

15. 100 READ C,D
    105 IF C = -999 THEN 999
    110 PRINT USING "**##.##    **##.##",C,D
    120 GOTO 100
    800 DATA 2.96,2.99,100,2345.93,3.42,10
    810 DATA -999,0
    999 END
```

```
16. 100 READ C,D
    105 IF C = -999 THEN 999
    110 PRINT USING "*###.##    *###.##",C,C
    120 GOTO 100
    800 DATA 2.96,2.99,100,2345.93,3.42,10
    810 DATA -999,0
    999 END

17. 100 READ C,D
    105 IF C = -999 THEN 999
    110 PRINT USING "###.##-    ###.##-",C,D
    120 GOTO 100
    800 DATA 24.26,29,26,123,25,-24.26,-29.26,-123.25
    810 DATA -999,0,0
    999 END

18. 100 READ C,D
    105 IF C = -999 THEN 999
    110 PRINT USING "####.##    ####.##",C,D
    120 GOTO 100
    800 DATA 24.26,29,26,123,25,-24.26,-29.26,-123.25
    810 DATA -999,0,0
    999 END

19. 100 READ A,B,C
    105 IF A = -999 THEN 999
    110 F$ = "### ### ###.##"
    120 PRINT USING F$,A,B,C
    130 GOTO 100
    800 DATA 200,300,10.22
    810 DATA -999,0,0
    999 END

20. 100 READ N$,N1$,A
    110 IF N$ = "LASTNAME" THEN 999
    120 F1$ = "\      \ \     \ IS ## YEARS OLD"
    130 PRINT USING F1$,N$,N1$,A
    140 GOTO 100
    800 DATA "ANNA","TAYLOR",45,"DONALD","BINZ",25,"ALBERT","DAVIS",3
    810 DATA "LASTNAME","L",0
    999 END
```

Assignment Sheet #5

1.
```
100 A$ = "**##,######.## DOLLARS"
110 INPUT "WHAT IS YOUR FIRST NAME"; F$
120 INPUT "WHAT IS YOUR MIDDLE NAME"; M$
130 INPUT "WHAT IS YOUR LAST NAME"; L$
140 INPUT "WHAT IS THE AMOUNT OF THE CHECK"; P
145 PRINT:PRINT
147 PRINT "PAY TO THE ORDER OF"
148 PRINT
150 PRINT USING "!. !. \                        \",F$, M$, L$
155 PRINT
160 PRINT USING A$,P
999 END
```

2.
```
100 REM  THIS PROGRAM IS USED TO PRINT THE TELEPHONE BILL AND
110 REM  TOTAL COST OF THE TELEPHONE BILL FOR EACH SIX MONTHS OF
120 REM  THE YEAR.  THE TOTAL YEARLY BILL, AVERAGE DAILY, AND
130 REM  AVERAGE MONTHLY COST IS ALSO PRINTED.
140 REM
150 REM     K = COUNTER FOR EACH HALF YEAR
155 REM     C = COUNTER FOR EACH MONTH
160 REM    T1 = TOTAL FOR FIRST HALF YEAR
165 REM    T2 = TOTAL FOR SECOND HALF YEAR
170 REM    A$ = MONTH
175 REM    A  = MONTHLY AMOUNT
180 REM    M  = MONTHLY AVERAGE     (YEARLY TOTAL/12)
185 REM    D  = DAILY AVERAGE       (YEARLY TOTAL/365)
190 REM
200 PRINT
210 PRINT TAB(3),"YEARLY TELEPHONE BILL"
220 PRINT
225 T2 = 0
230 FOR K = 1 TO 2
235 T1 = 0
240 FOR C = 1 TO 6
245 READ A$,A
250 T1 = T1 + A
260 PRINT USING "\         \   $#,###.##",A$,A
270 NEXT C
280 IF K = 1 THEN Y$ = "FIRST"
285 IF K = 2 THEN Y$ = "SECOND"
286 PRINT
290 PRINT USING "THE TOTAL FOR THE \    \ HALF OF THE YEAR IS
$##,###.##",Y$,T1
291 PRINT
300 PRINT
310 T2 = T2 + T1
320 NEXT K
```

```
330 M = T2/12
340 D = T2/365
350 PRINT USING "TOTAL TELEPHONE BILL FOR THE YEAR IS
$##,###.##",T2
355 PRINT
360 PRINT USING "AVERAGE DAILY AMOUNT IS $#,###.##",D
365 PRINT
370 PRINT USING "AVERAGE MONTHLY AMOUNT IS $#,###.##",M
375 PRINT
800 DATA "JANUARY",32.56,"FEBRUARY",34.66,"MARCH",55.55
801 DATA "APRIL",323.33,"MAY",32.56,"JUNE",32.56,"JULY",55.44
802 DATA "AUGUST",36.99,"SEPTEMBER",65.55,"OCTOBER",56.44
803 DATA "NOVEMBER",32.56,"DECEMBER",66.66
999 END
```

3.

```
100 REM  THIS PROGRAM CALCULATES DISCOUNT AND NET AMOUNTS FOR
110 REM  PURCHASES FROM ANDERSON APPLIANCE COMPANY.  THE
120 REM  DISCOUNTS ARE BASED UPON A CODE -  CODE 1 = 5%,
130 REM  CODE 2 = 10%, CODE 3 = 15%, AND CODE 4 = 20%.
140 REM
150 REM       N$ = CUSTOMER'S NAME
160 REM        C = CODE NUMBER
165 REM        A = AMOUNT OF PURCHASE
170 REM        D = AMOUNT OF DISCOUNT (A * DISCOUNT)
180 REM        N = NET AMOUNT (AMOUNT OF PURCHASE - DISCOUNT)
185 REM        F$, F1$, AND F2$ = FORMAT FIELDS FOR PRINT USING
200 PRINT
210 PRINT TAB(18)"ANDERSON APPLIANCE COMPANY"
220 PRINT
223 F1$= "  CUSTOMER'S        CODE      AMOUNT        AMOUNT
NET"
224 F2$= "     NAME         NUMBER    OF ORDER     OF DISCOUNT
AMOUNT"
225 F$ = "\              \     #      ##,###.##       #,###.##
##,###.##"
226 PRINT USING F1$
227 PRINT USING F2$
228 PRINT
230 READ N$,C,A
240 IF N$ = "LASTNAME" THEN 999
250 IF C = 1 THEN D = A * .05:GOTO 300
260 IF C = 2 THEN D = A * .10:GOTO 300
270 IF C = 3 THEN D = A * .15:GOTO 300
280 D = A * .20
300 N = A - D
310 PRINT USING F$, N$,C,A,D,N
320 GOTO 230
800 DATA "JANE ANDERSON",4,100,"ROBERT BALL",1,150
810 DATA "ANDREW CARMON",2,300, "SUSAN DREW",3,1000
820 DATA "TOM ELF",1,579,"BRENDA FELTS",2,199.98
820 DATA "MIKE GATES",2,235.66,"ANNE HIGGONS",4,2456.96
830 DATA "BRENDA ISOM",4,3456.78
850 DATA "LASTNAME",0,0
999 END
```

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

NAME ______________________

TEST

1. Match the terms on the right with the correct definitions.

_____a. Subroutines stored in the library of the computer's memory that are used in programs when special operations are to be performed

_____b. A valid BASIC arithmetic expression enclosed in parentheses with BASIC functions

_____c. A function which is used with the PRINT statement to tabulate to specific column positions

_____d. A special form of the PRINT statement that includes special features to print both numeric and string data

_____e. A statement that describes the arrangement used for a PRINT USING statement

_____f. Digits are dropped in contrast of rounding of numbers

_____g. In a PRINT USING statement, constants and variables are to be printed according to the format defined in the format field

_____h. In a PRINT USING statement, a set of character positions in which something is printed

1. Argument
2. Field
3. Format field
4. Function
5. List
6. PRINT USING
7. TAB function
8. Truncated

2. Select true statements concerning the PRINT USING statements by placing an "X" in the appropriate blanks.

_____a. The PRINT USING statement is useful for producing tables and reports with attractive appearance

_____b. The format field consists of only constants

_____c. One # in the format field is needed for each number to be printed _____d. A decimal point in the numeric format field causes a decimal number to round off when it extends past the # sign to the right of the decimal point

_____e. A % sign to the left of a number in the output indicates that too many spaces were formulated

_____f. If less numbers than # signs are to the right of the decimal point, blank spaces will be printed

_____g. Blank spaces are added to fill the spaces when fewer digits than # symbols are to the left of the decimal point

_____h. A comma can be placed in the print field to aid in readability of long numbers

_____i. No # symbol is required for a - sign to be printed in negative numbers

_____j. Either a \ or a % is used on most computers for string field format

_____k. If enough spaces are not allowed for the string data, the characters to the left of the formatted output will be truncated

_____l. A ! causes the first string character of the current value to be printed

_____m. Characters included in the quotation marks in the format field will not be printed

_____n. A $$ in the format field causes a floating $ sign

_____o. A single $ in the format field causes a floating $ sign

_____p. Double asterisks in the format field cause a floating asterisk

_____q. A format field can be used in an assignment (LET) statement to a string variable

_____r. If more variables or constants than # symbols are in the format field, the values will be continued on the next line using the same format on some computers

_____s. If fewer constants or variables than # symbols are in the format field, the extra fields will be filled with % signs

3. For each of the following PRINT USING statements, determine the output for the formatted fields. Use a b̸ to indicate blank spaces.

a.
```
10 PRINT USING "### ###"23,-46
```
____

b.
```
10 LET A = 10
20 LET B = 20
30 LET C = 2
40 PRINT USING"# ## ###",A,B,C
```
____

c.
```
10 LET A = 1000
20 LET B = 2000
30 PRINT USING "####.## ####.##",A,B
```
____

d.
```
10 PRINT USING"#,###  #,###",1000,2000
```
____

e.
```
10 PRINT USING "$$###",200
```
____

f.
```
10 PRINT USING "**######",2
```
____

g.
```
10 PRINT USING "$### $###",24,72
```
____

h.
```
10 LET A$ = ROBERT
20 LET B$ = SCOTT
30 LET C$ = GIBBONS
40 PRINT USING"\    \ \    \ \      \",A$,B$,C$
```
____

i.
```
10 LET A$ = ROBERT
20 LET B$ = SCOTT
30 LET C$ = GIBBONS
40 PRINT USING"!. !. \      \",A$,B$,C$
```
____

j.
```
10 LET N = 242
20 LET N$ = "ELM STREET"
30 PRINT USING "THE ADDRESS IS ### \          \",N,N$
```

____

4. Select true statements concerning the TAB function by placing an "X" in the appropriate blanks.

_____a. The TAB function is used to display large amounts of spaces

_____b. 10 PRINT TAB(20);"MAY" will cause MAY to print in column 21

_____c. A TAB function can be used in a PRINT USING statement

_____d. Constants, variables, and expressions are valid in TAB arguments

_____e. TAB function is very different from the TAB key of a typewriter

_____f. A semicolon and a comma after the parentheses in a TAB function produce the same results

_____g. When the value of the column number in the parentheses is less than or equal to the current position of the print line, the TAB is ignored

_____h. Arguments in TAB functions can be positive or negative values

_____i. If arguments in TAB functions have decimal points, an error message is printed

_____j. Only one TAB function is allowed per PRINT line

_____k. TAB functions are to be included in a PRINT USING statement

_____l. Arguments must be progressively smaller

_____m. When room between TABS is not supplied for the data to be printed, an error message will be printed

_____n. The TAB function is especially useful in printing headings for reports and tables

5. Code, enter, and RUN a program that uses different numeric format fields and values to produce formatted output.

6. Code, enter, and RUN a program that uses different string format fields and values to produce formatted output.

7. Code, enter, and RUN a program that uses three different string format fields and values to produce formatted output.

8. Code, enter, and RUN PRINT USING statements for selected output.

9. Flowchart, code, enter, and RUN programs using PRINT USING statements and TAB functions.

   (NOTE: If objectives 5 - 9 have not been accomplished prior to the test, ask your instructor when they should be completed.)

FORMATTED OUTPUT--PRINT USING STATEMENT AND TAB FUNCTION
UNIT VII

ANSWERS TO TEST

1. a. 4
   b. 1
   c. 7
   d. 6
   e. 3
   f. 8
   g. 5
   h. 2

2. a.
   c.
   d.
   g.
   h.
   j.
   i.
   l.
   n.
   p.
   q.
   r.

3. a. b̸23b̸-46
   b. $10b̸20b̸b̸b̸2
   c. 1000.00b̸2000.00
   d. 1,000b̸b̸2,000
   e. b̸$200
   f. *******2
   g. $b̸24b̸$b̸72
   h. ROBERTb̸b̸SCOTTb̸b̸GIBBONS
   i. R.b̸S.b̸GIBBONS
   j. THE ADDRESS IS 242 ELM STREET

4. a.
   b.
   d.
   g.
   n.

5. Evaluated to the satisfaction of the instructor.

6. Evaluated to the satisfaction of the instructor.

7. Evaluated to the satisfaction of the instructor.

8. Evaluated to the satisfaction of the instructor.

9. Evaluated to the satisfaction of the instructor.

FOR/NEXT STATEMENTS
UNIT VIII

UNIT OBJECTIVE

After completion of the unit, the student should be able to match terms associated with FOR/NEXT statements and select true statements concerning the FOR/NEXT statements; code a nested FOR/NEXT loop, and draw flowcharts from coded programs using a FOR/NEXT statement. The student should also be able to code, enter, and RUN simple programs using FOR/NEXT statements and flowchart, code, enter, and RUN programs using FOR/NEXT statements and nested FOR/NEXT statements. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with FOR/NEXT statements with the correct definitions.
2. Select true statements concerning FOR/NEXT statements.
3. Code a nested FOR/NEXT loop.
4. Draw flowcharts from coded programs using a FOR/NEXT statement in two flowcharting forms.
5. Determine the results of programs using the FOR/NEXT statements.
6. Code, enter, and RUN simple programs using FOR/NEXT statements.
7. Flowchart, code, enter, and RUN programs using FOR/NEXT statements and nested FOR/NEXT statements.

FOR/NEXT STATEMENTS
UNIT VIII

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

  A. Objective sheet

  B. Information sheet

  C. Transparency Master 1--Flowcharting FOR/NEXT statements

  D. Assignment sheets

    1. Assignment Sheet #1--Determine the results of programs using the FOR/NEXT statements

    2. Assignment Sheet #2--Code, enter, and RUN simple programs using FOR/NEXT statements

    3. Assignment Sheet #3--Flowchart, code, enter, and RUN programs using FOR/NEXT statements and nested FOR/NEXT statements

  E. Answers to assignment sheets

  F. Test

  G. Answers to test

II. References:

Refer to Unit I references.

FOR/NEXT STATEMENTS
UNIT VIII

INFORMATION SHEET

I. Terms and definitions

A. Initialization - setting a variable to a specific value

B. Incrementation - increasing in size

C. Decrementation - decreasing in size

D. FOR/NEXT statements - BASIC statements that direct the computer to repeat a set of instructions a specified number of times before control is transferred to the statements following the loop

E. FOR statement - BASIC statement that gives the assigned name of the control variable and sets its initial value, terminal value, and increment or decrement

F. Control variable - the variable used with FOR/NEXT statements to control the number of loops through a series of statements

G. Initial value - beginning value of the control variable of the FOR/NEXT loop

H. Terminal value - ending value of the control variable of the FOR/NEXT loop

I. NEXT statement - BASIC statement that marks the end of the loop

J. STEP option - sets the value of the increase or decrease of the control variable in a FOR/NEXT statement

K. Nested FOR/NEXT loops - the placement of a FOR/NEXT statement inside one or more FOR/NEXT statements

II. FOR/NEXT statements

A. Special looping structure for counter-controlled loops that directs the computer to repeat a set of instructions (loop) a specified number of times before control is transferred to the statements following the loop

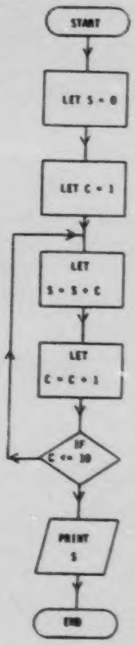
INFORMATION SHEET

B. Four parts of a loop

1. Initialization
2. Body of loop
3. Incrementation or decrementation
4. Testing for continuation or termination followed by unconditional transfer of control

Examples of loop with IF THEN statement:

```
                          10 S = 0
Initialization            20 C = 1
Body of loop              30 S = S + C
Incrementaton             40 C = C + 1
Testing and trans-        50 IF C <= 10 THEN 30
fer of control            60 PRINT "SUM OF INTEGERS 1
                          TO 10 =";S
                          99 END
```



Example of same program with FOR/NEXT statements:

```
10 S = 0
20 FOR C = 1 TO 10
30   S = S + C
40 NEXT C
50 PRINT "SUM OF INTEGERS 1 TO 10 =";S
99 END
```

INFORMATION SHEET

(NOTE: Initialization, incrementation, and testing and transfer of control are handled by the FOR/NEXT statements on lines 20 and 40.)

C. General format

1. FOR statement

```
                   10 FOR N = 1 TO 200 STEP 10
                    |  |  | | |  |  |    |   |
a.  Line number_____|  |  | | |  |  |    |   |
                       |  | | |  |  |    |   |
b.  The word FOR_______|  | | |  |  |    |   |
                          | | |  |  |    |   |
c.  The control variable  | | |  |  |    |   |
    (counter variable or  | | |  |  |    |   |
    index)________________| | |  |  |    |   |
                            | |  |  |    |   |
d.  The equal sign__________| |  |  |    |   |
                              |  |  |    |   |
e.  Initial value of the      |  |  |    |   |
    control variable__________|  |  |    |   |
                                 |  |    |   |
f.  The word TO__________________|  |    |   |
                                    |    |   |
g.  The terminal value of           |    |   |
    the control variable____________|    |   |
                                         |   |
h.  The word STEP________________________|   |
                                             |
i.  The STEP (increment or decrement) size___|
```

2. NEXT statement

```
                           50 NEXT N
                           |   |   |
a.  Line number____________|   |   |
                               |   |
b.  The word NEXT______________|   |
                                   |
c.  The same control variable      |
    name as in the corresponding   |
    FOR statement__________________|
```

3. The body of the loop is often indented 2 spaces to make the loop easier to read

```
10 FOR A = B TO C STEP D
20   LET S = S + A
30   PRINT S
40 NEXT A
99 END
```

INFORMATION SHEET

D. Purpose of FOR statement

1. Tells computer to begin loop
2. Assigns a numerical variable name to a control variable (index)
3. Defines initial (starting) value of counter
4. Defines terminal (ending) value of counter
5. Defines the STEP (increment or decrement) of the counter

E. Purpose of NEXT statement

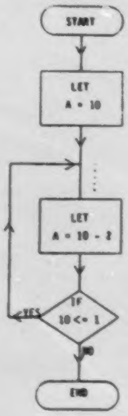
1. Represents the end of a loop that begins with a FOR statement
2. The computer adds the STEP value to the control variable when the NEXT statement is encountered
3. The computer then checks to see if the variable is less than or equal to the terminal value of the counter
4. If variable exceeds the terminal value, the loop is complete and control is transferred to the statement following the NEXT statement

   (NOTE: If STEP option is negative, the loop ends when the counter is less than the final value.)

   Example:

```
10 FOR A = 10 TO 1 STEP -2
.
.
.
20 NEXT A
```

   Loop is complete when A <= 1



5. If variable is less than or equal to terminal value, control is transferred to the beginning of the loop again

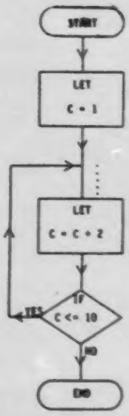
INFORMATION SHEET

F. Rules

1. Each and every FOR statement must have one and only one NEXT statement corresponding to it
2. The line number of the FOR statement must be lower than the line number of the NEXT statement
3. Counter variable must be a numeric variable
4. Variable name in NEXT statement must be the same as the one in the FOR statement
5. Initial value, final value, and increment may be:

    a. Constants

```
10 FOR C = 1 TO 10 STEP 2
.
.
.
40 NEXT C
```



    b. Variables

```
10 FOR C = A TO B STEP X
.
.
.
80 NEXT C
```

    c. Arithmetic expressions

```
10 FOR X = A TO B*C STEP (T*C/D+3)
.
.
.
60 NEXT X
```

INFORMATION SHEET

d. Positive numbers

```
10 FOR J = 1 TO 10 STEP 5
.
.
.
80 NEXT J
```

e. Negative numbers

```
10 FOR J = -1 TO 15 STEP 3
.
.
.
80 NEXT J
```

OR

```
10 FOR N = 10 TO -1 STEP -1
.
.
.
50 NEXT N
```

f. Noninteger number

```
10 FOR C = 5 TO 15 STEP .5
.
.
.
70 NEXT C
```

g. Combination of the above

```
10 FOR N = A TO (3*A)/X STEP 2
.
.
.
50 NEXT N
```

6. When STEP option is not included in the FOR statement, the default of STEP +1 is used

Example:

```
10 LET C = 1 TO 200
```

is the same as

```
10 LET C = 1 TO 200 STEP 1
```

INFORMATION SHEET

(NOTE: STEP value should not be zero. This will cause an infinite loop.)

7. The value of counter variables must not be changed by any statement within the loop area or the loop will not operate properly

8. The value of the counter variable may be used by any statement inside the loop area

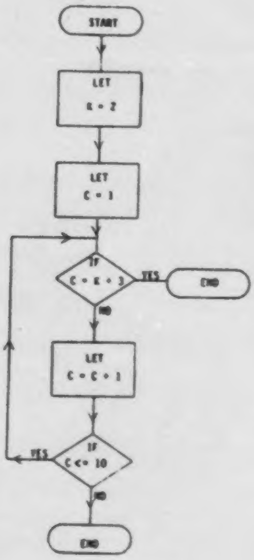
   Example:

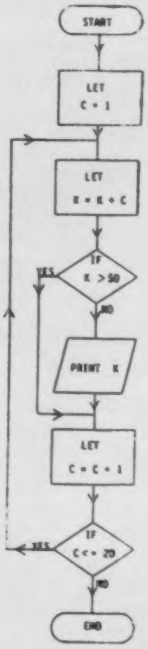
```
10 FOR J = 1 TO 10
20   LET S = S + J
30   PRINT "LOOP";J;"TOTAL IS";S
40 NEXT J
```

9. The program can exit from a FOR/NEXT loop at any point with a GO TO statement or an IF THEN statement to a statement out of the loop or to the NEXT statement

   Examples:

```
10 LET K = 2                  10 FOR C = 1 TO 20
20 FOR C = 1 TO 10            20   K = K + C
30   IF C = K + 3 THEN 99     30   IF K > 50 THEN 50
40 NEXT C                     40   PRINT K
99 END                        50 NEXT C
                              99 END
```


START
LET
K = 2
LET
C = 1
IF
C = K + 3
YES
END
NO
LET
C = C + 1
IF
C <= 10
YES
NO
END


START
LET
C = 1
LET
K = K + C
IF
K > 50
YES
NO
PRINT K
LET
C = C + 1
IF
C <= 20
YES
NO
END

INFORMATION SHEET

10. The program should branch to a FOR/NEXT loop only at the FOR statement

    Example:

```
20 IF A = 10 THEN 40
30 ...
40 FOR C = 1 TO 10
50   LET A = A + C
60 NEXT C
70 ...
99 END
```

    (NOTE: A programmer should not transfer control to inside a FOR/NEXT loop. This may cause errors to result.)

11. When initial and final values are equal and the STEP is nonzero, the loop is executed only once

G. Variation on control variable value at termination of loop

1. Variable retains last value in loop

   a. On some computers, the value of the counter variable will be equal to the last value of the counter variable when the loop terminates

   b. The value can then be used later outside of the loop in the program

   Example:

```
10 FOR C = 1 TO 100
20   LET T = T + C
30 NEXT C
40 LET A = T/C
50 PRINT AVERAGE =";A
99 END
```

2. Variable does not retain last value in loop value

   a. On other computers, the value will be greater than the final counter variable value

   b. The value cannot be used outside the loop and obtain accurate results

INFORMATION SHEET

(NOTE: It is safer not to depend on the variable retaining its last value unless the instruction manual states that the last value is retained.)

H. Two ways of determining initial loop execution of FOR/NEXT statements:

1. On some computers, the statements within the loop will execute one time even if the terminal value is more than the initial value

2. On other computers, the initial value of the counter will be checked with the final value before the loop is processed; if the final value exceeds the initial value, the loop will not be executed at all

   (NOTE: This can be tested by entering and running the following program on your computer:

```
10 FOR C = 10 TO 1
20   PRINT C
30 NEXT C
99 END
```

   The program will print the number 10 if the computer executes the loop once. No number will be printed if the computer checks the initial and final value before the loop is executed.)

III. Nested FOR/NEXT statements

A. Any number of loops can be contained within an initial loop

B. The inner (nested) FOR/NEXT loop must be completely contained within the outer FOR/NEXT loop

Example:

Proper nesting

```
10 FOR I = 1 TO 10
20 FOR J = 1 TO 15
30 NEXT J
40 NEXT I
```

Improper nesting

```
10 FOR I = 1 TO 10
20 FOR J = 1 TO 15
30 NEXT I
40 NEXT J
```

C. Control passes through the inner loop each time it passes through the outer loop

D. If loops are crossed, the program will not run

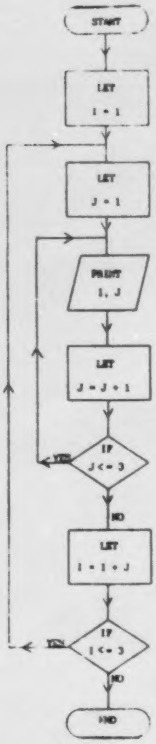
INFORMATION SHEET

Example:

```
10 FOR I = 1 TO 5
20   FOR J = 1 TO 3
30     PRINT I,J
40   NEXT J
50 PRINT
60 NEXT I
99 END
```

```
RUN

 1             1
 1             2
 1             3
 2             1
 2             2
 2             3
 3             1
 3             2
 3             3
 4             1
 4             2
 4             3
 5             1
 5             2
 5             3
```

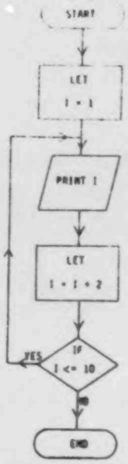

START
LET
I = 1
LET
J = 1
PRINT
I, J
LET
J = J + 1
IF
J <= 3
YES
NO
LET
I = I + J
IF
I <= 3
YES
NO
END

IV. Flowcharting FOR/NEXT statements (Transparency 1)

A. Flowcharted in same way the initialization, increment, and test of IF/THEN statement is flowcharted

Example:

```
10 FOR I = 1 TO 10 STEP 2
20   PRINT I;
30 NEXT I
99 END
```

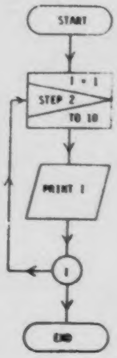

START
LET
I = 1
PRINT I
LET
I = I + 2
IF
I <= 10
YES
NO
END

INFORMATION SHEET

B. Flowcharted by use of a specially designed flowcharting symbol that combines the process and decision box and uses a connector

Example:

```
10 FOR I = 1 TO 10 STEP 2
20    PRINT I;
30 NEXT I
99 END
```

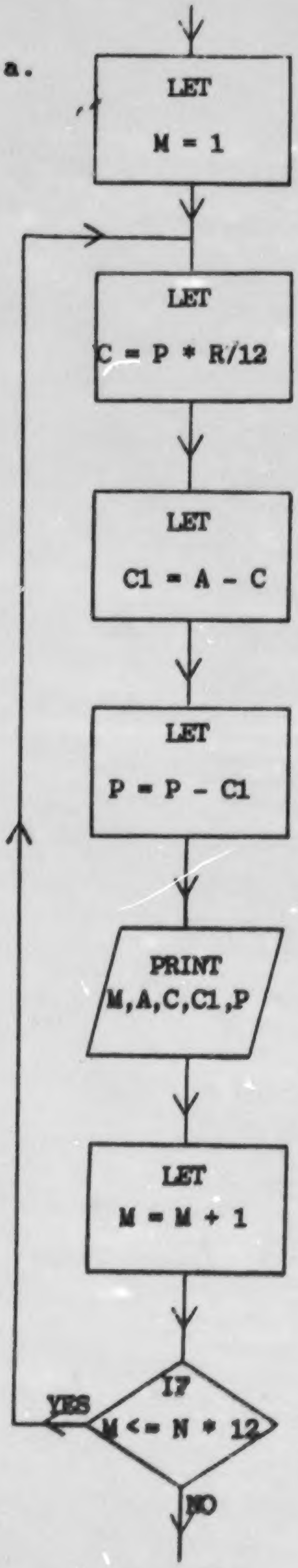

START
I = 1
STEP 2
TO 10
PRINT I
I
END

```
100 REM  THIS PROGRAM CALCULATES THE MONTHLY PAYMENT OF A
110 REM  MORTGAGE.  IT ALSO PRINTS OUT THE PAYMENT NUMBER,
120 REM  THE PAYMENT, THE INTEREST PAID EACH MONTH, THE AMOUNT
130 REM  THE PRINCIPAL IS REDUCED EACH MONTH, AND THE BALANCE
140 REM  OF THE PRINCIPAL.
150 REM
160 REM         P = PRINCIPAL BALANCE
170 REM         R = INTEREST RATE
180 REM         N = NUMBER OF YEARS TO PAY OUT LOAN
190 REM         A = AMOUNT OF MONTHLY PAYMENT
200 REM         M = CONTROL VARIABLE FOR FOR/NEXT LOOP
210 REM         C = CURRENT INTEREST PAID EACH MONTH
220 REM         C1= CURRENT REDUCTION OF PRINCIPAL
230 REM             EACH MONTH
240 REM         F$, F1$, AND F2$ = FORMAT FIELDS FOR
250 REM         PRINT USING STATEMENTS
300 INPUT "ENTER PRINCIPAL (AMOUNT OF LOAN)";P
310 INPUT "ENTER INTEREST RATE";R
320 INPUT "ENTER NUMBER OF YEARS OF LOAN";N
325 R = R/100
330 A = P * (R/12)/(1-(1/(1+(R/12))**(N*12)))
340 PRINT
350 F$="PAYMENT    MONTHLY     INTEREST     PRINCIPAL
PRINCIPAL"
355 F1$=" NUMBER    PAYMENT       PAID          PAID
BALANCE"
360 PRINT
370 PRINT USING F$
375 PRINT USING F1$
380 F1$ = "  ###      #####.##     ####.##       ####.##
#####.##
400 FOR M = 1 TO N*12
410 C= P * R/12
420 C1 = A - C
430 P = P - C1
440 PRINT USING F1$,M,A,C,C1,P
450 NEXT M
999 END
```

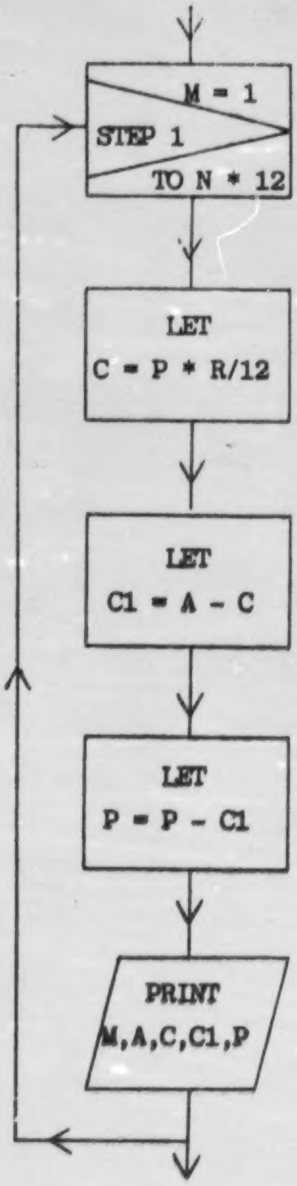
The flowchart of the FOR/NEXT loop for the program is shown flowcharted in two ways on the next page.

TM 1

a.


LET
M = 1
LET
C = P * R/12
LET
C1 = A - C
LET
P = P - C1
PRINT
M,A,C,C1,P
LET
M = M + 1
IF
M <= N * 12
YES
NO

b.


M = 1
STEP 1
TO N * 12
LET
C = P * R/12
LET
C1 = A - C
LET
P = P - C1
PRINT
M,A,C,C1,P

FOR/NEXT STATEMENTS
UNIT VIII

ASSIGNMENT SHEET #1--Determine the results of programs using the FOR/NEXT statements

Directions: Determine the printed results from the following segments of programs using FOR/NEXT statements:

1.
```
10 FOR A = 1 TO 4
20   LET X = A
30 NEXT A
40 PRINT X
```

2.
```
10 LET C = 0
20 FOR B = 1 TO 5
30  LET C = C+B
40 NEXT B
50 PRINT C
```

3.
```
10 LET C = 0
20 FOR B = 1 TO 5 STEP 2
30  LET C = C+B
40 NEXT B
50 PRINT C
```

4.
```
10 LET K = 6
20 FOR I = 1 TO 6
30   LET K = K-I
40 NEXT I
50 PRINT K
```

5.
```
10 LET Z = 0
20 FOR C = 4 TO 1 STEP -1
30   LET Z = Z + C
40 NEXT C
50 PRINT Z
```

6.
```
10 LET L = 6
20 FOR J = 1 TO 10 STEP 3
30  LET L = L + J
40 NEXT J
50 PRINT L
```

7.
```
10 FOR C = 1 TO 9 STEP 2
20   LET A = C - 1
30 NEXT C
40 PRINT A
```

8.
```
10 LET X = 1
20 FOR C = X TO X
30  PRINT C
40 NEXT C
```

ASSIGNMENT SHEET #1

9.
```
10 FOR C = -5 TO 5
20  PRINT C
30 NEXT C
```

10.
```
10 FOR C = 3 TO 6 STEP 3
20  PRINT C
30 NEXT C
```

11.
```
10 FOR A = 6 TO 1 STEP -1
20   FOR B = 1 TO 5
30     PRINT A;B
40   NEXT B
50  PRINT
60 NEXT A
```

12.
```
10 FOR C = 1 TO 10 STEP 2
20   PRINT C + 1
30 NEXT C
```

13.
```
10 LET A = 5
20 LET B = 2
30 LET C = 15
40 FOR D = A TO C STEP B
50  PRINT D;A;C;B
60 NEXT D
```

14.
```
10 FOR C = 1 TO 3 STEP .1
20   PRINT C
30 NEXT C
```

15.
```
10 FOR C = 10 TO 10
20  PRINT C
30 NEXT C
```

ASSIGNMENT SHEET #1

```
9.  10 FOR C = -5 TO 5
    20  PRINT C
    30 NEXT C

10. 10 FOR C = 3 TO 6 STEP 3
    20  PRINT C
    30 NEXT C

11. 10 FOR A = 6 TO 1 STEP -1
    20   FOR B = 1 TO 5
    30     PRINT A;B
    40   NEXT B
    50  PRINT
    60 NEXT A

12. 10 FOR C = 1 TO 10 STEP 2
    20   PRINT C + 1
    30 NEXT C

13. 10 LET A = 5
    20 LET B = 2
    30 LET C = 15
    40 FOR D = A TO C STEP B
    50  PRINT D;A;C;B
    60 NEXT D

14. 10 FOR C = 1 TO 3 STEP .1
    20   PRINT C
    30 NEXT C

15. 10 FOR C = 10 TO 10
    20  PRINT C
    30 NEXT C
```

FOR/NEXT STATEMENTS
UNIT VIII

ASSIGNMENT SHEET #2--Code, enter, and RUN simple programs using FOR/NEXT statements

1. Write a FOR/NEXT statement which will repeat a PRINT statement 10 times.

2. Write a FOR/NEXT statement which will repeat a math calculation four times and print the total at the end of the loop.

3. Write a FOR/NEXT statement which will repeat a mathematical calculation 10 times and print the result of each loop.

4. Write a FOR/NEXT statement which will repeat a mathematical calculation 10 times with STEP 2. Print the result of each loop.

5. Write a FOR/NEXT statement which will cause execution of two FOR/NEXT statements in sequence.

   The outer loop is executed three times
   The inner loop two times

   Print out the values of each control variable within the inner loop.

6. Write a FOR/NEXT statement which will cause execution of three FOR/NEXT statements in sequence.

   The outer loop is executed three times
   The first inner loop two times
   The second inner loop four times

   Print out the values of each control variable.

7. Code, enter, and RUN a program using a FOR/NEXT statement to sum the odd integers from 1 to 100.

8. Code, enter, and RUN a program using a FOR/NEXT statement to sum the even integers from 2 to 100.

ASSIGNMENT SHEET #2

9. The following sample program prints the letter N:

```
10 FOR I = 1 TO 9
20 PRINT "*";TAB(I);"*";TAB(10);"*"
30 NEXT I
99 END
```

```
**        *
* *       *
*  *      *
*   *     *
*    *    *
*     *   *
*      *  *
*       * *
*        **
```

Code, enter, and RUN a program to print the letter H

```
*         *
*         *
*         *
*         *
***********
*         *
*         *
*         *
*         *
```

10. Code, enter, and RUN a program to print the letter P

```
****
*   *
*    *
*   *
****
*
*
*
*
```

FOR/NEXT STATEMENTS
UNIT VIII

ASSIGNMENT SHEET #3--Flowchart, code, enter, and RUN programs using FOR/NEXT statements and nested FOR/NEXT statements

1. Flowchart, code, enter, and RUN a program using a FOR/NEXT statement to allow the user to input 10 numbers ranging from -99 to 99. Assign the numbers as variable A and variable B. Add A plus B and multiply A times B. Print the answers in the following format:

```
10.00      +      20.00      =      30.00
10.00      X      20.00      =     200.00
```

2. Flowchart, code, enter, and RUN a program using FOR/NEXT statements that allows the user to determine the number of grades to be entered; enter the grades, print the highest grade, print the lowest grade, and print the grade average. Read the first grade and assign it as the high and low grade before entering the FOR/NEXT loop.

3. Flowchart, code, enter, and RUN a program using nested FOR/NEXT statements to read 6 grades for 5 students. The first 4 grades count 15 points each. The last 2 grades count 20 points each. Print out the student's name and the total points.

Leah McCrotty	15	13	15	13	20	16
Lauren McKnight	13	12	10	15	20	10
David Neaves	15	15	14	15	19	20
Joyce Oudegeest	10	15	12	13	17	16
Sharon Short	9	10	10	5	9	6

4. Flowchart, code, enter, and RUN a program using a FOR/NEXT statement to convert inches to centimeters. Print a table for the conversion of 1 to 20 inches. Round the centimeters to 2 decimal places.

   1 inch = 2.54 centimeters

5. Flowchart, code, enter, and RUN a program using a FOR/NEXT statement to print out a multiplication table from 0 to 12. Use an INPUT statement to enter the value of the multiplication table desired.

6. Flowchart, code, enter, and RUN a program using FOR/NEXT statements to print a table showing compound interest earned on varying amounts of money at varying interest rates.

   Three FOR/NEXT statements will be needed for this program.

ASSIGNMENT SHEET #3

Use the following formula to calculate the interest earned:

A = P*(1+R)**T
A = amount paid in interest over the set time period
P = principal at beginning of first time period
R = interest rate
T = number of time periods

Amount of money from $500 to $2000 in $500 increments.

Interest rate from 8% to 12% in 1% increments.

Time of investment from 5 to 20 years in increments of 5.

Print out the amount of money, the rate, the time, the interest earned, and the accumulated amount.

7. Flowchart, code, enter, and RUN a program using nested FOR/NEXT statements to read values of 10 items. Calculate the sum and the average of the 10 values. Use values of 100,150,175,200,250, 100,110,120,140,200

   Print the values using the PRINT USING statement.

8. Flowchart, code, enter, and RUN a program using a FOR/NEXT statement to print a chart that shows the amounts earned for working 40 hours at a rate of $3.50 to $12.50 per hour at $.25 increments. Include both the rate per hour and the total amount earned for the 40 hours. Use a PRINT USING statement to print the amounts with $ signs and 2 decimal places.

Hourly Rate	Total Amount
$3.50	$140.00

9. Flowchart, code, enter, and RUN a program using a FOR/NEXT statement, to compute the payroll of the Solutions Company. The company employs 20 people. Compute their gross pay.

   Use the following data:

A. BLOOM	4.50	35	B. CARTER	5.00	4
C. DODGE	3.60	20	D. EARL	7.50	40
E. FREEMAN	10.00	25	F. GATES	4.00	10
G. HUDGINS	5.50	40	H. INGRAM	8.25	36
I. JONES	9.50	20	J. KENT	5.50	34
K. LOVE	4.75	34	L. MANNING	3.60	30
M. NORRIS	6.50	29	N. OZMOND	5.60	40
O. POOLE	6.75	40	P. QUARTZ	3.50	40
Q. RODGERS	12.50	40	R. SMITH	4.50	24
S. THOMAS	6.00	40	T. URBANKEE	5.00	40

10. Flowchart, code, enter, and RUN a program to produce a bar chart for the following data:

Sales for Company 1	20,000
Sales for Company 2	50,000
Sales for Company 3	35,000
Sales for Company 4	40,000

Produce the bars by using the = sign and a FOR/NEXT loop for sales for each company

Base the bar graph on sales per thousand

Print appropriate headings

11. Flowchart, code, enter, and RUN a program using a FOR/NEXT statement to determine if an employee is eligible for retirement. The following criteria is needed to be eligible for retirement:

Must be 65 years of age or older

Must be more than 60 years of age and worked at least 10 years

Must be at least 52 years old and worked for 25 years or over

FOR/NEXT STATEMENTS
UNIT VIII

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

1. 4
2. 15
3. 9
4. -15
5. 10
6. 28
7. 8
8. 1
9.
```
-5
-4
-3
-2
-1
 0
 1
 2
 3
 4
 5
```
10.
```
3
6
```
11.
```
6  1
6  2
6  3
6  5

5  1
5  2
5  3
5  4
5  5

4  1
4  2
4  3
4  4
4  5
```

```
        3   1
        3   2
        3   3
        3   4
        3   5

        2   1
        2   2
        2   3
        2   4
        2   5

        1   1
        1   2
        1   3
        1   4
        1   5

12.     2
        4
        6
        8
        10

13.     5   5   15   2
        7   5   15   2
        9   5   15   2
        11   5   15   2
        13   5   15   2
        15   5   15   2

14.    1
       1.1
       1.2
       1.3
       1.4
       1.5
       1.6
       1.7
       1.8
       1.9
       2
       2.1
       2.2
       2.3
       2.4
       2.5
       2.6
       2.7
       2.8
       2.9
       3

15.    10
```

Assignment Sheet #2

1.
```
100 FOR A = 1 TO 10
110   PRINT A
120 NEXT A
999 END
```

2.
```
100 FOR A = 1 TO 4
110   A1 = A1 + A
120 NEXT A
130 PRINT A1
999 END
```

3.
```
100 FOR A = 1 TO 10
110  A1 = A * 2
120   PRINT A1
130 NEXT A
999 END
```

4.
```
110 FOR A = 1 TO 20 STEP 2
120   A1 = A1 + A
125 PRINT A
130 NEXT A
999 END
```

5.
```
100 FOR A = 1 TO 3
110   FOR B = 1 TO 2
120     PRINT A,B
130   NEXT B
140 NEXT A
999 END
```

6.
```
100 FOR A = 1 TO 3
110   FOR B = 1 TO 2
120     FOR C = 1 TO 4
130       PRINT A,B,C
140     NEXT C
150   NEXT B
160 NEXT A
999 END
```

7.
```
100 FOR I = 1 TO 100 STEP 2
110   S = S + I
120 NEXT I
130 PRINT "THE SUM OF THE ODD INTEGERS FROM 1 TO 100 IS";S
999 END
```

8.
```
100 FOR I = 2 TO 100 STEP 2
110  S = S + I
120 NEXT I
130 PRINT "THE SUM OF THE EVEN INTEGERS FROM 1 TO 100 IS";S
999 END
```

9.
```
10 FOR A = 1 TO 4
20 PRINT "*";TAB(10);"*"
30 NEXT A
40 FOR B = 1 TO 9
50 PRINT "*";
60 NEXT B
70 FOR C = 1 TO 5
80 PRINT "*";TAB(10);"*"
90 NEXT C
99 END
```

10.
```
110 FOR A = 1 TO 4
120 PRINT "*";
130 NEXT A
140 PRINT
150 PRINT "*";TAB(4);"*"
160 PRINT "*";TAB(5);"*"
170 PRINT "*";TAB(4);"*"
180 FOR B = 1 TO 4
190 PRINT "*";
200 NEXT B
210 PRINT
220 FOR C = 1 TO 4
230 PRINT "*"
240 NEXT C
250 PRINT
999 END
```

Assignment Sheet #3

1.
```
100 REM  THIS PROGRAM ALLOWS THE USER TO INPUT 2 NUMBERS
110 REM  BETWEEN -99 AND 99 FROM THE TERMINAL.  THE
120 REM  PROGRAM WILL THEN ADD AND MULTIPLY THE 2 NUMBERS
130 REM  AND PRINT THE ANSWERS
140 REM
150 REM      A = FIRST NUMBER ENTERED
160 REM      B = SECOND NUMBER ENTERED
170 REM      T = TOTAL FOR ADDITION
180 REM      P = TOTAL FOR MULTIPLICATION
185 REM      I = COUNTER FOR FOR/NEXT LOOP
190 REM
200 FOR I = 1 TO 10
105 PRINT
110 READ A,B
115 IF A < -99 OR A > 99 THEN PRINT "NUMBERS MUST BE BETWEEN
-99 AND 99 - TRY AGAIN":GOTO 110
116 IF B < -99 OR B > 99 THEN PRINT "NUMBERS MUST BE BETWEEN
-99 AND 99 - TRY AGAIN":GOTO 110
120 T = A + B
125 P = A * B
130 PRINT USING "### + ### = #####",A,B,T
135 PRINT USING "### X ### = #####",A,B,P.
140 NEXT I
999 END
```

2.

```
100 REM  THIS PROGRAM ALLOWS THE USER TO INPUT THE NUMBER OF
110 REM  GRADES TO BE USED, SELECT THE HIGH AND LOW GRADE,
120 REM  CALCULATE TOTAL POINTS AND THE AVERAGE GRADE.
130 REM
140 REM     N = NUMBER OF GRADES TO BE ENTERED
150 REM     G = EACH GRADE
160 REM     I = COUNTER FOR LOOP
170 REM     T = TOTAL POINTS
180 REM     A = AVERAGE GRADE
190 REM
200 INPUT "HOW MANY GRADES ARE YOU GOING TO ENTER";N
210 INPUT G
220 H = G
230 L = G
235 T = 0
240 FOR I = 2 TO N
245 T = T + G
250 INPUT G
260   IF G > H THEN H = G
270   IF G < L THEN L = G
280 NEXT I
300 A = T/N
310 PRINT USING "THE HIGH GRADE IS ###.##",H
320 PRINT USING "THE LOW GRADE IS ###.##",L
330 PRINT USING "TOTAL POINTS = #####.##",T
340 PRINT USING "AVERAGE GRADE =  ###.##",A
999 END
```

3.

```
100 REM  THIS PROGRAM PRINTS THE NAME OF 5 STUDENTS AND THEIR
110 REM  TOTAL POINTS
120 REM
130 REM       N$ = STUDENT NAME
140 REM       G  = GRADES
150 REM       T  = TOTAL POINTS
160 REM    L,M   = LOOP CONTROL COUNTERS
170 REM
200 FOR L = 1 TO 5
210   READ N$
215   T = 0
220   FOR M = 1 TO 6
230     READ G
240     T = T + G
250   NEXT M
260   PRINT USING "\             \      ####.## POINTS",N$,T
300 NEXT L
800 DATA "LEAH MCROTTY",15,13,15,13,20,16
801 DATA "LAUREN MCKNIGHT",13,12,10,15,20,10
802 DATA "DAVID NEAVES",15,15,14,15,19,20
803 DATA "JOYCE OUDEGEEST",10,15,12,13,17,16
804 DATA "SHARON SHORT",9,10,10,5,9,6
999 END
```

```
4.  100 PRINT "INCHES     CENTIMETERS"
    110 FOR I = 1 TO 20
    120   LET C = I * 2.54
    130   PRINT USING " ###        ###.##",I,C
    140 NEXT I
    150 END

5.  100 REM  THIS PROGRAM ALLOWS THE USER TO SELECT A
    110 REM  MULTIPLICATION TABLE FROM 0 TO 12.  THE
    120 REM  TABLE WILL THEN BE PRINTED.
    130 REM      N = MULTIPLICATION TABLE TO PRINT
    140 REM      I = LOOP COUNTER
    150 REM      P = PRODUCT TO BE PRINTED
    160 REM
    200 INPUT "WHICH MULTIPLICATION TABLE DO YOU WISH TO PRINT";N
    205 PRINT
    210 FOR I = 0 TO 12
    220   P = I * N
    230   PRINT USING "## X ## = ###",I,N,A
    240 NEXT I
    999 END

6.  90 F$="#####.##      ##.##      ##      #####.##"
    100 FOR P = 500 TO 5000 STEP 500
    110   FOR R = .08 TO .12
    120       FOR T = 5 TO 20 STEP 5
    130         A = P*(1+R)**T
    140         PRINT USING F$,P,R,T,A
    150       NEXT T
    160      PRINT
    170    NEXT R
    180   PRINT
    190 NEXT P
    999 END

7.  100 REM  THIS PROGRAM READS 10 NUMBERS, TOTALS THE NUMBER,
    110 REM  AND CALCULATES THE AVERAGE OF THE 10 NUMBERS.  THE
    120 REM  AMOUNTS ARE THEN PRINTED
    125 REM
    130 REM     N = NUMBER
    140 REM     T = TOTAL
    150 REM     A = AVERAGE
    160 REM     I = LOOP CONTROL
    170 REM
    200 FOR I = 1 TO 10
    210   READ N
    220   T = T + N
    230 NEXT I
    240 A = T/10
    250 PRINT USING "SUM = #,###     AVERAGE = ###.##",T,A
    800 DATA 100,150,175,200,250,100,110,120,140,200
    999 END
```

```
8.  100 REM  THIS PROGRAM PRINTS THE AMOUNT EARNED FOR WORKING 40
    110 REM  HOURS PER WEEK AT THE RATE OF $3.50 - $12.50 PER
    120 REM  HOUR IN INCREMENTS OF $.25
    130 REM
    140 REM        A = AMOUNT EARNED
    150 REM        I = LOOP CONTROL FOR RATES OF $3.50 - $12.50
    160 REM
    200 PRINT USING "HOURLY          TOTAL"
    205 PRINT
    210 FOR I = 3.5 TO 12.5 STEP .25
    220   A = I * 40
    230   PRINT USING "$##.##     $#,###.##",I,A
    240 NEXT I
    999 END

9.  100 REM  THIS PROGRAM COMPUTES GROSS PAY FOR 20 EMPLOYEES.
    110 REM  THE EMPLOYEE'S NAME, RATE OF PAY, HOURS WORKED, AND
    120 REM  GROSS PAY IS PRINTED.
    130 REM
    140 REM        N$ = EMPLOYEE'S NAME
    150 REM        R  = RATE OF PAY
    160 REM        H  = HOURS WORKED
    170 REM        G  = GROSS PAY
    180 REM        I  = LOOP CONTROL VARIABLE
    190 REM
    200 PRINT USING "      EMPLOYEE          RATE   HOURS   GROSS
    PAY"
    210 PRINT
    215 FOR I = 1 TO 20
    220   READ N$,R,H
    230   G = R * H
    240   PRINT USING "\                    \    $##.##    ##
    $###.##",N$,R,H,G
    250 NEXT I
    800 DATA "A. BLOOM",4.50,35,"B. CARTER",5.00,40,"C.
    DODGE",3.60,20
    810 DATA "D. EARL",7.50,40,"E. FREEMAN",10.00,25,"F.
    GATES",4.00,10
    820 DATA "G. HUDGINS",5.50,40,"H. INGRAM",8.25,36
    830 DATA "I. JONES",9.50,20,"J. KENT",5.50,34,"K.
    LOVE",4.75,34
    840 DATA "L. MANNING",3.60,30,"M. NORRIS",6.50,29
    850 DATA "N. OZMOND",5.60,40,"O. POOLE",6.75,40,
    860 DATA "P. QUARTZ",3.50,40,"Q. RODGERS",12.50,40
    870 DATA "R. SMITH",4.50,24,"S. THOMAS",6,40,"T.
    URBANKEE",5.00,40
    999 END
```

10.

```
100 REM  THIS PROGRAM PRINTS A BAR GRAPH FOR SALES OF 4
110 REM  DEPARTMENTS.  EACH "=" REPRESENTS $1000 IN SALES.
120 REM
130 REM        S   = SALES BY DEPARTMENTS
140 REM        I,B = LOOP CONTROL VARIABLS
150 REM
200 PRINT
205 PRINT "                 EACH EQUAL SIGN REPRESENTS $1000
OF SALES"
206 PRINT
210 FOR I = 1 TO 4
220   READ S
230   PRINT "COMPANY";I,
240   FOR B = 1 TO S/1000
250     PRINT "=";
260   NEXT B
270   PRINT
280 NEXT I
800 DATA 20000,50000,35000,40000
999 END
```

11.

```
100 REM  THIS PROGRAM DETERMINES IF AN EMPLOYEE IS ELIGIBLE
110 REM  FOR RETIREMENT USING THE FOLLOWING CRITERIA:
120 REM  65 YEARS OF AGE OR OLDER, MORE THAN 60 YEARS OF AGE
130 REM  AND WORKED AT LEAST 10 YEARS, AT LEAST 52 YEARS OLD
140 REM  AND WORKED FOR 25 YEARS OR OVER
150 REM   E = LOOP CONTROL VARIABLE
160 REM   A = AGE
170 REM  N$ = NAME
180 REM   Y = YEARS WORKED
200 FOR E = 1 TO 10
210 READ N$,A,Y
220 IF A>= 65 THEN 300
230 IF A >= 60 AND Y >= 10 THEN 300
240 IF A >=52 AND Y >= 25 THEN 300
250 PRINT N$; " IS NOT ELIGIBLE FOR RETIREMENT"
255 PRINT
260 GO TO 310
300 PRINT N$; " IS ELIGIBLE FOR RETIREMENT"
305 PRINT
310 NEXT E
400 DATA "GLENDA ATKINS",60,5,"CARMA BECKER",40,20,"STEVE
CAMPBELL",25,10
410 DATA "SCOTT SIMPSON",65,4,"BRENDA TYRA",53,27,"PHILIP
WATSON",61,11
420 DATA "JOYCE WILLIAMS",64,15,"DEBORAH YANCEY",27,5,"JAY
TURNER",63,5
430 DATA "TREY VELVIN",52,25
999 END
```

FOR/NEXT STATEMENTS
UNIT VIII

NAME______________________________

TEST

1. Match the terms on the right with the correct definitions.

_____a. Increasing in size

_____b. BASIC statement that gives the assigned name of the control variable and sets its initial value, terminal value, and increment or decrement

_____c. BASIC statements that direct the computer to repeat a set of instructions a specified number of times before control is transferred to the statements following the loop

_____d. Decreasing in size

_____e. Setting a variable to a specific value

_____f. The placement of a FOR/NEXT statement inside one or more FOR/NEXT statements

_____g. Ending value of the control variable of the FOR/NEXT loop

_____h. Sets the value of the increase or decrease of the control variable in a FOR/NEXT statement

_____i. BASIC statement that marks the end of the loop

_____j. Beginning value of the control variable of the FOR/NEXT loop

_____k. The variable used with FOR/NEXT statements to control the number of loops through a series of statements

1. Control variable
2. Decrementation
3. FOR/NEXT statements
4. FOR statement
5. Incrementation
6. Initial value
7. Initialization
8. Nested FOR/NEXT loops
9. NEXT statement
10. STEP option
11. Terminal value

2. Select true statements concerning the FOR/NEXT statements by placing an "X" in the appropriate blanks.

_____a. The FOR statement signals the end of the FOR/NEXT loop _____b. If the STEP option is not included in the FOR statement, the computer prints an error message

_____c. The initial value is the starting value

_____d. The terminal value is the increment or decrement value

_____e. It is not necessary to have a NEXT statement if the loop ends at the end of the program

_____f. Counter variables must be a numeric variable

_____g. All computers retain the terminal value of the loop after the loop is ended

_____h. A loop will always be processed at least once

_____i. A STEP size of 0 will produce an infinite loop

_____j. A program should branch to a FOR/NEXT loop only at the NEXT statement

_____k. A program can exit from a FOR/NEXT loop at any point by using a GO TO statement or an IF/THEN statement

_____l. The value of the control variable cannot be used inside the loop area

_____m. It is acceptable to change the value of the control variable within a loop

_____n. When the initial and final values of the FOR statement are equal, the loop will be executed once

_____o. Inital values, final values, and STEP values may be constants, variables, arithmetic expressions, and/or positive or negative integer and/or noninteger numbers

_____p. The line number for the NEXT statement must be lower than the line number for the FOR statement

_____q. When the NEXT statement is encountered, the computer adds the STEP value to the control variable and checks to see if the counter variable is less than or equal to the terminal value

_____r. When the control variable exceeds the terminal value, the loop returns to the NEXT statement under the FOR statement _____s. The four parts of a loop are: 1) initialization, 2) body of loop, 3) incrementation or decrementation, and 4) testing for continuation or termination followed by the unconditional transfer of control

_____t. The body of a FOR/NEXT loop must not be indented or an error message will be printed

3. Code a nested FOR/NEXT loop to produce the following output:

```
FIRST LOOP
  SECOND LOOP
      THIRD LOOP
      THIRD LOOP
  SECOND LOOP
      THIRD LOOP
      THIRD LOOP
FIRST LOOP
  SECOND LOOP
      THIRD LOOP
      THIRD LOOP
  SECOND LOOP
      THIRD LOOP
      THIRD LOOP
```

4. Draw the flowcharts for the following program. Draw the first flowchart using separate process and decision boxes. Draw the second flowchart using the combined process and decision box and a connector.

```
10 FOR I = 1 TO 200 STEP 2
20   PRINT I
30     FOR J = 2 TO 10
40       PRINT I;J
50     NEXT J
60   PRINT
70 NEXT I
99 END
```

5. Determine the results of programs using the FOR/NEXT statements.

6. Code, enter, and RUN simple programs using FOR/NEXT statements.

7. Flowchart, code, enter, and RUN programs using FOR/NEXT statements and nested FOR/NEXT statements.

(NOTE: If objectives 5 - 7 have not been accomplished prior to the test, ask your instructor when they should be completed.)

FOR/NEXT STATEMENTS
UNIT VIII

ANSWERS TO TEST

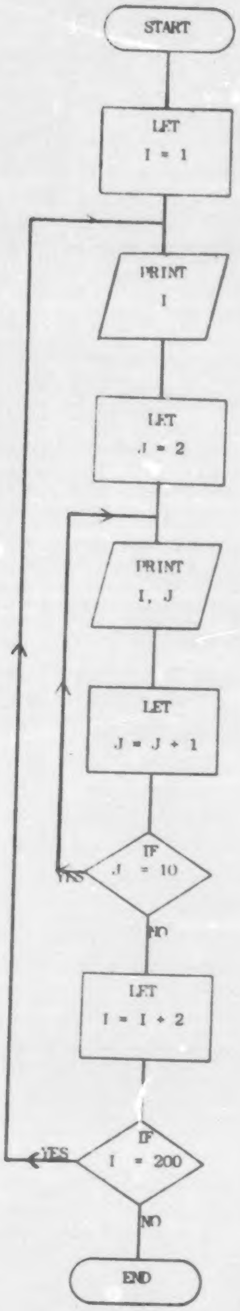
1. a. 5
   b. 4
   c. 3
   d. 2
   e. 7
   f. 8
   g. 11
   h. 10
   i. 9
   j. 6
   k. 1

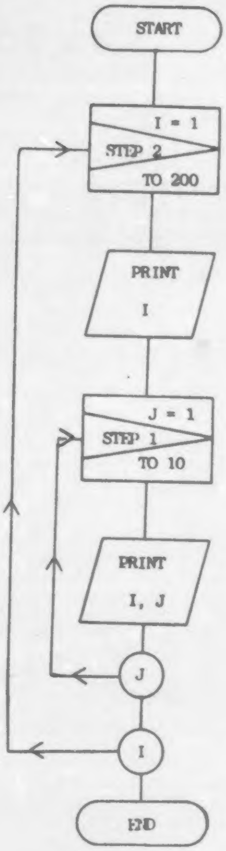
2. f.
   i.
   k.
   m.
   n.
   q.
   r.
   s.

3.

```
100 FOR R1 = 1 TO 2
110   PRINT "FIRST LOOP"
120   FOR R2 = 1 TO 2
130     PRINT "   SECOND LOOP"
140     FOR R3 = 1 TO 2
150     PRINT "      THIRD LOOP"
160     NEXT R3
170   NEXT R2
180 NEXT R1
999 END
```

4.


START
LET
I = 1
PRINT
I
LET
J = 2
PRINT
I, J
LET
J = J + 1
IF
J = 10
YES
NO
LET
I = I + 2
IF
I = 200
YES
NO
END


START
I = 1
STEP 2
TO 200
PRINT
I
J = 1
STEP 1
TO 10
PRINT
I, J
J
I
END

5. Evaluated to the satisfaction of the instructor.

6. Evaluated to the satisfaction of the instructor.

7. Evaluated to the satisfaction of the instructor.

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

UNIT OBJECTIVE

After completion of this unit, the student should be able to match terms associated with built-in functions and subroutines and select true statements concerning built-in functions and subroutines. The student should also be able to write BASIC expressions containing built-in functions; write the results of built-in functions; determine the output of programs containing subroutines; and flowchart, code, enter, and RUN programs using built-in functions and subroutines. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with built-in functions and subroutines with their correct definitions.
2. Select true statements concerning built-in functions.
3. Select true statements concerning subroutines.
4. Write BASIC statements containing built-in functions.
5 Write the results of built-in functions.
6. Determine the output of programs containing subroutines.
7. Flowchart, code, enter, and RUN programs using built-in functions and subroutines.

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

   A. Objective sheet

   B. Information sheets

   C. Assignment sheets

      1. Assignment Sheet #1--Write BASIC statements containing built-in functions

      2. Assignment Sheet #2--Write the results of built-in functions

      3. Assignment Sheet #3--Determine the output of programs containing subroutines

      4. Assignment Sheet #4--Flowchart, code, enter, and RUN programs using built-in functions and subroutines

   D. Answers to assignment sheets

   E. Test

   F. Answers to test

II. References:

Refer to Unit I references.

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

INFORMATION SHEET

I. Terms and definitions

A. Built-in (library) functions - subroutines stored in the library of the computer's memory that are used in programs when special operations are to be performed

B. Argument - a valid arithmetic expression enclosed in parentheses that is used with BASIC functions

C. Function - automatic subroutines

D. Seed - a data item used to begin a series of calculations

E. Random sequence - an ordered set of numbers, each of which may not be predicted from any knowledge of its predecessors

F. Subroutine - series of statements that constitute a program that can be executed only when "called" by another program

G. GOSUB statement - BASIC statement used to transfer control to subroutines ("call" subroutines)

H. RETURN statement - BASIC statement used to transfer control back to the main program

I. INT (integer) function - returns the value of the greatest integer that is less than or equal to the argument

J. SGN (sign) function - returns the value of the sign only

K. ABS (absolute value) function - returns the absolute value of the argument (determines how far the argument is from 0)

L. RND (random number) function - generates random numbers between 0 and 1

M. SQR (square root) function - returns square root of any positive value

INFORMATION SHEET

II. Built-in functions

A. Built-in functions (library functions) are subroutines that are stored in the computer's memory that are used when special operations are to be performed

B. Calculates the single resulting value of an argument

C. General format

```
100 INT(X)
```

1. Line number
2. The three unique identifying letters of the function
3. The argument enclosed in parentheses

D. Arguments can be

1. Single variables

   Example:

   ```
   INT(A)
   ```

2. Single constants

   Example:

   ```
   INT(3.14)
   ```

3. Any valid mathematical expression

   Example:

   ```
   INT(A*N**C/3)
   ```

E. Common functions

1. SQR (square root) function

   a. Returns square root of any positive value

INFORMATION SHEET

Example:

```
10 FOR X = 25 TO 50 STEP 5
20    S = SQR(X)
30    PRINT "THE SQUARE ROOT OF";X;"IS";S
40 NEXT X
99 END

RUN

THE SQUARE ROOT OF 25 IS 5
THE SQUARE ROOT OF 30 IS 5.47723
THE SQUARE ROOT OF 35 IS 5.91608
THE SQUARE ROOT OF 40 IS 6.32456
THE SQUARE ROOT OF 45 IS 6.70823
THE SQUARE ROOT OF 50 IS 7.07107
```

b. If argument is negative, an error message will be printed

2. INT (integer) function

   a. Returns the value of the greatest integer that is less than or equal to the argument

   b. If argument is positive, returned value is the integer of the argument

   Example:

```
10 PRINT INT(3.1455)
99 END

RUN

 3
```

   c. If argument is negative, returned value is the integer preceding the argument

   Example:

```
10 PRINT INT(-3.1455)
99 END

RUN

-4
```

   d. To round off numbers using the INT function, .5 is added to the argument

INFORMATION SHEET

Example:

10 X = INT(X + .5) will round X to the nearest whole number

e. To round off numbers to 2 decimal places using the INT function, multiply the argument by 100 and then divide the INT result by 100

Example:

```
10 X = INT(X*100+.5)/100
```

3. SGN (sign) function

    a. Returns the value of the sign only

    b. Three possible values returned:

        1) When X > 0, value returned is 1

        2) When X = 0, value returned is 0

        3) When X < 0, value returned is -1

Example:

```
10 FOR X = -2 TO 2 STEP .5
20    S = SGN(X)
30 PRINT USING "THE SIGN OF ##.# IS
##",X,S
40 NEXT X
99 END
```

```
RUN

THE SIGN OF -2.0 IS -1
THE SIGN OF -1.5 IS -1
THE SIGN OF -1.0 IS -1
THE SIGN OF -0.5 IS -1
THE SIGN OF  0.0 IS  0
THE SIGN OF  0.5 IS  1
THE SIGN OF  1.5 IS  1
THE SIGN OF  2.0 IS  1
```

4. ABS (absolute value) function

    a. Returns the absolute value of the argument

    b. Determines how far argument is from 0

INFORMATION SHEET

c. Always makes argument positive or 0

  1) When X > 0, value will be X

  2) When X = 0, value will be 0

  3) When X < 0, value will be X without the - sign

  Example:

```
10 FOR X = -1 TO 1 STEP .5
20   A = ABS(X)
30   PRINT USING "ABSOLUTE VALUE OF
##.# IS ##.#",X,S
40 NEXT X
99 END
```

```
RUN

THE ABSOLUTE VALUE OF -1.0 IS  1.0
THE ABSOLUTE VALUE OF -0.5 IS  0.5
THE ABSOLUTE VALUE OF  0.0 IS  0.0
THE ABSOLUTE VALUE OF  0.5 IS  0.5
THE ABSOLUTE VALUE OF  1.0 IS  1.0
```

5. RND (random number) function

  a. Generates random numbers between 0 and 1

  b. Generates a 6-digit fractional number

  c. Often used in games and simulations

  d. Rules for using the RND function vary extensively on computers

  (NOTE: See your instruction manual for more detailed operating instructions.)

    1) TRS-80 rules

      a) RND(0) produces different 6-digit random numbers each time the program is executed

      Example:

```
10 FOR X = 1 TO 5
20   PRINT RND(0)
30 NEXT X
99 END
```

INFORMATION SHEET

```
RUN

 .204935
 .229581
 .533074
 .132211
 .995602
```

(NOTE: Any 6-digit fractional number greater than 0 and less than 1 will appear each time the program is executed. The numbers will not be the same.)

b) RND(X) when X is a positive integer value between 1 and 32768, random integer numbers up to the value of the argument are generated

Example:

10 X = RND(70) generates random integers greater than 0 and less than 71

c) To produce random integer numbers between a range of numbers beginning with a number other than 1, use the following formula and the INT function:

```
INT((VALUE RANGE +1)*RND(0) + LOW
VALUE)
```

Example:

To produce random integer values between 50 and 85, use:

```
10 X = INT(36*RND(0)+50)
```

d) To produce two-place decimals, divide the formula given above by 100

Example:

To produce random two-place decimals between .50 and .75, use:

```
10 X = INT(26*RND(0)+50)/100
```

INFORMATION SHEET

e) To generate random positive and negative numbers, use the formula:

INT((VALUE RANGE + 1)*RND(0) - LOW VALUE)

Example:

To generate integers between -30 and 30, use:

```
10 X = INT(61 * RND(0) - 30)
```

f) The statement RANDOM can be used at the beginning of the program to ensure an unpredictable sequence of random numbers each time the program is used

Example:

```
10 RANDOM
20 FOR I = 1 TO 3
30   PRINT RND(10)
40 NEXT I
99 END
```

(NOTE: The TRS-80 does produce the exact same random numbers each time the program is executed without the RANDOM statement. Some computers, however, do not. The RANDOM statement does ensure that the numbers are seeded again before the program is RUN.)

2) PDP 11/34 (RSTS/E operating system) rules

a) RND without an argument is used to produce random 6-digit fractional numbers greater than 0 and less than 1

Example:

```
10 FOR I = 1 TO 3
20   PRINT RND
30 NEXT I
99 END
```

INFORMATION SHEET

```
RUN

 .23455
 .99959
 .52222
```

b) The same random numbers will be printed unless the RANDOMIZE or RANDOM statement is used at the beginning of the program

Example:

```
10 FOR I = 1 TO 3
20   PRINT RND
30 NEXT I
99 END

RUN

 .23455
 .99959
 .52222

RUN

 .23455
 .99959
 .52222

10 RANDOMIZE
20 FOR I = 1 TO 3
30   PRINT RND
40 NEXT I
99 END

RUN

 .345666
 .678999
 .122333

RUN

 .145543
 .890432
 .002123
```

c) To produce random integer numbers between 1 and 32768, use the INT function and multiply the RND function by the largest number needed plus 1

INFORMATION SHEET

Example:

To generate numbers greater than 0 and less than or equal to 6, use:

```
20 A = INT(7 * RND)
```

(NOTE: The number 0 will not be included.)

d) To produce random integer numbers between a range of numbers beginning with a number other than 1, use the following formula:

```
INT((VALUE RANGE +1)*RND + LOW
VALUE)
```

Example:

To produce random integer values between 50 and 85, use:

```
10 X = INT(36*RND + 50)
```

e) To produce a two-decimal random number, divide the formula given in c by 100

Example:

To produce random two-place decimals between .50 and .75, use:

```
10 X = INT(26*RND+50)/100
```

f) To generate random positive and negative numbers, use the formula:

```
INT((VALUE RANGE + 1) * RND - LOW
VALUE)
```

Example:

To generate random positve and negative numbers between -30 and 30, use:

```
10 X = INT(61 * RND -30)
```

INFORMATION SHEET

(NOTE: Other functions are available on most computers. See your instruction manual for more information on these.)

III. Subroutines

- A. A subroutine is a series of statements that constitute a program that can be executed only when "called" by another program
- B. Computer transfers control back and forth between the main program and the subroutine(s)
- C. Purpose
    - 1. Used when a group of statements need to be used several times in a program
    - 2. Makes program easier to write, read, and understand when written in smaller units
    - 3. Saves space
    - 4. Decreases chance of programming errors
    - 5. Simplifies the program structure
- D. Two statements required
    - 1. GOSUB statement
        - a. Transfers control to the subroutine beginning at the specified line number of the GOSUB statement
        - b. Stores an address to return to after the subroutine is complete
        - c. When computer encounters the RETURN statement, it returns control to the statement which follows the GOSUB statement
    - 2. RETURN statement
        - a. Ends a subroutine
        - b. Returns control to the statement immediately following the most recently executed GOSUB statement

INFORMATION SHEET

E. General format

1.
```
                                    100 GOSUB 200
                                     |    |    |
a.  Line number______________________|    |    |
                                          |    |
b.  The word GOSUB________________________|    |
                                               |
c.  The line number designating                |
    the beginning of the subroutine____________|
```

2.
```
                                    400 RETURN
                                     |    |
a.  Line number______________________|    |
                                          |
b.  The word RETURN_______________________|
```

Example:

```
100 GOSUB 200
110 GOSUB 400
120 GOSUB 300
130 GOTO 999
200 PRINT "LINE 1"
210 RETURN
300 PRINT "LINE 3"
310 RETURN
400 PRINT "LINE 2"
410 RETURN
999 END

RUN

LINE 1
LINE 2
LINE 3
```

F. Procedure

1. Stops at line number of GOSUB
2. Goes to line number specified in GOSUB statement
3. Executes lines in subroutine until a RETURN statement is executed
4. Returns the control of the program to the line immediately following the last executed GOSUB statement

INFORMATION SHEET

Example:

```
100   A = 5
110   B = 36
120   D = B/3
130 GOSUB 180
140 PRINT A;B;C;D;E
150   H = A + B
160 PRINT H
170 GO TO 999
180   C = B - A
190   E = C + D
200 RETURN
999 END

RUN

 5 36 31 12 43
 41
```

G. Rules

1. A main program may contain one or more subroutine
2. Subroutines can be used as many times as needed
3. The statement numbers in subroutines cannot conflict with those in the main program
4. Any valid BASIC statement can be used in a subroutine
5. Subroutines can be located anywhere in the program but it is a good idea to place them just before the END statement
6. Transfer of control to subroutines must be made by a GOSUB statement
7. Transfer of control back to main program must be made by a RETURN statement
8. More than one RETURN statement may be used if multiple RETURN conditions are needed
9. The first line of a subroutine should be a REM statement explaining what the subroutine does
10. The subroutine must be separated from the last line of the main program so the subroutine will not be executed when it is not appropriate

INFORMATION SHEET

H. Nested subroutines

1. A subroutine can call another subroutine
2. A subroutine cannot call itself
3. Control is returned to the line immediately following the last executed GOSUB statement that calls the subroutine

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

ASSIGNMENT SHEET #1--Write BASIC statements containing built-in functions

Directions: Write a BASIC statement containing built-in functions for each of the following results. Assign the result to the variable R.

1. The square root of 25
2. The integer value of -3.4567
3. The integer value of 45678.33
4. The integer value of the square root of 2
5. The rounded-off integer value of 5.6983
6. The integer value of 156.789 rounded to two decimal places
7. The sign value of 145.66
8. The sign value of -2983.22
9. The sign value of 0
10. The absolute value of -456.33

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

ASSIGNMENT SHEET #2--Write the results of built-in functions

Directions: For problems 1 - 12 in Assignment Sheet #1, write the results of the built-in functions in the blank spaces provided below:

1. ________________________________

2. ________________________________

3. ________________________________

4. ________________________________

5. ________________________________

6. ________________________________

7. ________________________________

8. ________________________________

9. ________________________________

10. ________________________________

11. ________________________________

12. ________________________________

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

ASSIGNMENT SHEET #3--Determine the output of programs containing subroutines

Directions: On the lines provided, write the output for the programs containing subroutines and nested subroutines.

1.
```
100 GOSUB 200
110 GOSUB 300
120 GOSUB 200
130 GOTO 999
200 PRINT "PLEASE"
210 RETURN
300 PRINT "SAY"
310 RETURN
999 END
```

____

____

2.
```
100 LET A = 1
120 LET B = 5
130 LET C = 1
140 GOSUB 200
145 GOSUB 300
150 LET A = 5
160 LET B = 1
170 LET C = -1
180 GOSUB 300
190 GOSUB 200
195 GOTO 999
200 REM FIRST SUBROUTINE
210 PRINT TAB(30);"*"
220 RETURN
300 REM SECOND SUBROUTINE
310 FOR X = A TO B STEP C
320   PRINT TAB(30-X);"*";TAB(30+X);"*"
330 NEXT X
340 RETURN
999 END
```

____

____

3.
```
100 READ A,B,D
110 GOSUB 200
120 GOSUB 400
130 GOSUB 300
140 PRINT A,B,C,D
150 GOTO 999
```

ASSIGNMENT SHEET #3

```
200 REM FIRST SUBROUTINE
210 LET C = A + B
220 GOSUB 300
230 RETURN
300 REM SECOND SUBROUTINE
310 LET D = D + 1
330 RETURN
400 REM THIRD SUBROUTINE
410 PRINT A,B,C,D
420 RETURN
800 DATA 1,2,3
999 END
```

____

____

4.
```
100 LET C = 3
110 GOSUB 400
120 PRINT A,B,C
200 LET C = 6
210 GOSUB 400
220 PRINT A,B,C
230 N = A + B + 3 + C
240 PRINT N
250 GOTO 999
400 A = C * C**2
410 B = C * C * C*2
420 C = C * C * C/3
430 RETURN
999 END
```

____

____

5.
```
100 LET A = 5
120 LET B = 3.6
130 LET D = A + B
140 GOSUB 200
150 PRINT A;B;D;F;G
160 LET H = F + G
170 PRINT H
180 GO TO 999
200 LET F = A + B
210 LET G = B + D
200 RETURN
999 END
```

____

____

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

ASSIGNMENT SHEET #4--Flowchart, code, enter, and RUN programs using built-in functions and subroutines

Directions: For the following problems, flowchart, code, enter, and RUN programs using built-in functions and/or subroutines to produce the desired results.

1. Flowchart, code, enter, and RUN a program that will print 10 names, addresses, and telephone numbers in three possible formats.

   The formats are:

   1. Name only

   2. Name and address

   3. Name and telephone number

   Three separate subroutines are needed for this program.

2. Flowchart, code, enter, and RUN a program that will ask the user of the program to input 10 numbers (five sets of numbers), two numbers at a time. The program will then ask the user to enter the sum and the difference of the two numbers. Subroutines are to be used to keep track of the number of right and wrong answers.

   An immediate response will be given after each answer for correct and incorrect answers. If the answer is correct print "CORRECT - ENTER 2 MORE NUMBERS". If the answer is incorrect, print "INCORRECT ANSWER - THE CORRECT ANSWER IS ____ - TRY 2 MORE NUMBERS"

3. Using the RND function, flowchart, code, enter, and RUN a program to simulate the flipping of a coin 50 times. Keep track of the number of heads and tails that occur. Heads will be 0. Tails will be 1.

4. Rewrite program 2 to use the RND function to provide the two numbers between 10 and 50.

5. Flowchart, code, enter, and RUN a program to randomly select five winning numbers for raffle tickets from 001 to 500.

ASSIGNMENT SHEET #4

6. Flowchart, code, enter, and RUN a program to randomly select numbers. The user will be asked to guess the numbers. Ten tries will be allowed. If the user guesses too high, a message should be printed "TOO HIGH - TRY AGAIN". If the guess is too low, "TOO LOW - TRY AGAIN" should be printed. After the tenth try, the correct number should be printed and the user should be asked if he would like to try again.

7. Flowchart, code, enter, and RUN a program to ask the user to input five numbers one at a time. The program will then determine if the number is even or odd using the integer statement. Hint: You will need to divide the number inputted by 2 and check to see if it is equal to the integer of the number inputted.

8. Flowchart, code, enter, and RUN a program to compute the amount of sales tax for each purchase and the total of the purchase plus the sales tax. Compute the tax at a rate of 3% for Mississippi, 4% for Arkansas, 5% for Texas, and 6% for California.

   Use an INPUT statement to enter the amount of each purchase.

   Use a subroutine to perform the calculations and print each purchase, tax, and total amount for each state.

9. Flowchart, code, enter, and RUN a program to:

   READ employee name, hourly wage, hours worked each day of the week (Saturday through Sunday)

   Use the following data:

Name	Rate	Hours Per Week						
		Sun	Mon	Tues	Wed	Thurs	Fri	Sat
Dale Finley	$4.00	8	10	6	8	4	5	6
Judy Swanson	5.00	2	5	8	8	8	8	8
Nathan Hart	3.75	0	4	4	4	4	4	0
Troy Mander	7.50	0	8	8	8	8	8	0
Mary Franklin	8.00	2	8	8	10	8	8	0
Anthony Stewart	3.60	0	6	6	6	6	6	0
Mathew Dumas	10.00	0	8	8	8	8	4	4
Kathy Munson	12.00	1	8	10	10	8	8	5
Michael Johnson	5.00	0	5	5	5	6	7	0
Timothy Richards	4.50	2	4	4	4	8	3	2

   The program will print the hours worked, the amount earned by the employee on each day of the week, the total hours worked, and the total amount earned for the week.

ASSIGNMENT SHEET #4

The company pays time-and-a-half for Saturday work, double-time for Sunday work, and time-and-a-half for all hours over 40 on Monday - Friday.

Use subroutines to calculate earnings on Monday - Friday.

Your output should be similar to the following:

```
TOM LIGHT     $4.00 PER HOUR

 DAY            HOURS           AMOUNT EARNED

 SUNDAY           0             $000.00
 MONDAY           8             $ 32.00
 TUESDAY          4             $ 16.00
 WEDNESDAY       10             $ 40.00
 THURSDAY        10             $ 40.00
 FRIDAY          10             $ 40.00
 SATURDAY         2             $ 12.00

 OVERTIME         2             $  4.00

 TOTALS

 HOURS WORKED    44
 TOTAL EARNINGS                 $184.00


 OVERTIME IS COMPUTED ON THE FOLLOWING INFORMATION

 TOTAL HOURS WORKED MONDAY - FRIDAY      42

 OVERTIME ADDITIONAL PAY FOR 2 HOURS      4
```

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

1. R = SQR(25)
2. R = INT(-3,4567)
3. R = INT(45678.33)
4. R = INT(SQR(2))
5. R = INT(5.6983 + .5)
6. R = INT(156.789 * 100 + .5)/100
7. R = SGN(145.66)
8. R = SGN(-2983.22)
9. R = SGN(0)
10. R = ABS(-456.33)
11. R = ABS(567.34)
12. R = ABS(0)
13. R = RND(0) OR R = RND
14. R = RND(20) OR R = INT(RND *21)
15. R = INT (101 * RND(0)) + 100
16. R = INT(91 * RND(0))-40
17. RANDOMIZE OR RANDOM
18. R = INT (51 * RND(0) + 10)/100
19. R = ABS(INT(SQR(5)))
20. R = SGN(INT(SQR(144)))

Assignment Sheet #2

1. 5
2. -4
3. 45678
4. 1
5. 6
6. 156.79
7. 1
8. -1
9. 0
10. 456.33
11. 567.34
12. 0

Assignment Sheet #3

1.
```
PLEASE
SAY
PLEASE
```

2.
```
                             *
                            * *
                           *   *
                          *     *
                         *       *
                        *         *
                        *         *
                         *       *
                          *     *
                           *   *
                            * *
                             *
```

3.
```
 1  2  3  1
 1  2  3  2
```

4.
```
 27              54               9
 216             432              72
 723
```

5.
```
 5  3  8  8  1
 19
```

Assignment Sheet #4

1.
```
01 REM  THE FOLLOWING PROGRAM PRINTS NAMES, ADDRESSES,
02 REM  AND TELEPHONE NUMBERS IN THREE DIFFERENT FORMATS:
03 REM     NAMES ONLY
04 REM     NAMES AND ADDRESSES
05 REM     NAMES AND TELEPHONE NUMBERS
06 REM     N   = VARIABLE NAME FOR SELECTED SUBROUTINE
07 REM     I   = LOOP CONTROL VARIABLE
08 REM     N$  = NAME
09 REM     A$  = ADDRESS
10 REM     T$  = TELEPHONE NUMBER
11 REM
15 PRINT " ENTER THE NUMBER 1 IF YOU WANT TO PRINT OUT
```

```
NAME ONLY"
20 PRINT " ENTER THE NUMBER 2 IF YOU WANT TO PRINT OUT
NAME AND ADDRESS"
30 PRINT " ENTER THE NUMBER 3 IF YOU WANT TO PRINT OUT
NAME AND TELEPHONE NUMBER"
40 PRINT
42 F$ = "\                     \      \                   \"
50 INPUT "ENTER NUMBER";N
60 IF N<1 OR N>3 THEN PRINT "SELECT NUMBERS 1, 2, OR 3
ONLY":GOTO 50
65 PRINT
70 IF N=1 THEN GOSUB 100
80 IF N=2 THEN GOSUB 200
90 IF N=3 THEN GOSUB 300
95 GOTO 999
100 REM SUBROUTINE TO PRINT NAMES ONLY
105 PRINT " NAME"
106 PRINT
110 FOR I=1 TO 10
120   READ N$,A$,T$
130   PRINT USING F$,N$
140 NEXT I
150 RETURN
200 REM  SUBROUTINE TO PRINT NAMES AND ADDRESSES
205 PRINT " NAME                  ADDRESS"
206 PRINT
210 FOR I=1 TO 10
220   READ N$,A$,T$
230   PRINT USING F$,N$,A$
240 NEXT I
250 RETURN
300 REM SUBROUTINE TO PRINT NAMES AND TELEPHONE NUMBERS
305 PRINT " NAME            TELEPHONE NUMBER"
310 FOR I=1 TO 10
320   READ N$,A$,T$
330   PRINT USING F$,N$,T$
340 NEXT I
350 RETURN
890 DATA "KEN ALLISON","PARK STREET","632-7543"
900 DATA "PAULA WELCH","ELK STREET","224-9999"
910 DATA "SUE BATES","PARK STREET","376-9876"
920 DATA "JOE ADAMS","GATE COVE","999-9876"
930 DATA "FRANK THOMAS","ROCK STREET","865-9543"
940 DATA "TOM HAYNIE","BRADFORD COURT","929-2423"
950 DATA "NEWTON SMITH","IZARD","272-7374"
960 DATA "DANA DUNN","KINGS PARK ROAD","914-8148"
970 DATA "ALICE SOUTH","28TH STREET","274-1471"
980 DATA "LOUIS NORTH","MAIN STREET","231-1433"
999 END
```

2.

```
010 REM  THIS PROGRAM ALLOWS THE USER TO INPUT 5 SETS OF
020 REM  NUMBERS.  THE USER THEN ENTERS THE ANSWER FOR
030 REM  ADDITION AND SUBTRACTION OF THE 2 NUMBERS.  THE
040 REM  CORRECT ANSWER IS PRINTED IF THE ANSWER IS
```

```
050 REM  INCORRECT.  THE TOTAL CORRECT AND INCORRECT
055 REM  ANSWERS ARE PRINTED AT THE END OF THE PROGRAM.
060 REM  N = FIRST NUMBER ENTERED
065 REM  N1= SECOND NUMBER ENTERED
070 REM  I = LOOP CONTROL VARIABLE
072 REM  A = ANSWER ENTERED BY USER
075 REM  C = TOTAL OF CORRECT ANSWERS
080 REM  W = TOTAL OF INCORRECT ANSWERS
085 REM  C1= CORRECT ANSWER
090 PRINT "THIS PROGRAM WILL ALLOW YOU TO ENTER 5 SETS OF
2 NUMBERS EACH."
091 PRINT "YOU WILL THEN COMPUTE THE ANSWERS FOR ADDITION
AND SUBTRACTION."
092 PRINT
99 FOR I = 1 TO 5
100 INPUT "PLEASE ENTER TWO NUMBERS WITH COMMAS
BETWEEN",N,N1
110 PRINT "PLEASE ENTER THE ANSWER FOR";N;"+";N1;"=";
120 INPUT A
122 C1 = N + N1
125 IF A = C1 THEN GOSUB 250:GOTO 140
130 GOSUB 300
140 PRINT "PLEASE ENTER THE ANSWER FOR";N;"-";N1;"=";
150 INPUT A1
155 C1 = N - N1
160 IF A1 = C1 THEN GOSUB 250:GOTO 180
170 GOSUB 300
180 NEXT I
200 PRINT USING "OUT OF 10 PROBLEMS, YOU ANSWERED ##
CORRECT AND ## INCORRECT",C,W
210 GOTO 999
250 REM SUBROUTINE FOR CORRECT ANSWERS
260 PRINT "YOUR ANSWER IS CORRECT"
270 C = C + 1
280 RETURN
300 REM SUBROUTINE FOR INCORRECT ANSWERS
310 PRINT " INCORRECT ANSWER "
320 PRINT "THE CORRECT ANSWER IS ";C1
330 W = W + 1
340 RETURN
999 END
```

3.

```
100 REM THIS PROGRAM SIMULATES THE FLIPPING OF A COIN 50
110 REM TIMES.  THE TOTAL OF HEADS AND TAILS IS PRINTED
120 REM AT THE END OF THE PROGRAM.
130 REM  I = LOOP CONTROL VARIABLE
140 REM  C = VARIABLE FOR HEADS OR TAILS
150 REM  T = TOTAL FOR TAILS
160 REM  H = TOTAL FOR HEADS
170 RANDOMIZE
200 FOR I = 1 TO 50
210   C = INT(RND(0) + .5)
220   IF C = 1 THEN T = T + 1
230   IF C = 0 THEN H = H + 1
240 NEXT I
```

```
250 PRINT "THE COIN LANDED ON HEADS";H;"TIMES"
260 PRINT "THE COIN LANDED ON TAILS";T;"TIMES"
999 END
```

4.

```
010 REM THIS PROGRAM ALLOWS THE USER TO ANSWER ADDITION
020 REM AND SUBTRACTION PROBLEMS FOR 5 SETS OF RANDOM
025 REM NUMBERS.  THE USER ENTERS THE ANSWER FOR ADDITION
030 REM AND SUBTRACTION OF THE 2 NUMBERS.  THE CORRECT
040 REM ANSWER IS PRINTED IF THE ANSWER IS INCORRECT.  THE
050 REM TOTAL CORRECT AND INCORRECT ANSWERS ARE PRINTED AT
055 REM THE END OF THE PROGRAM.
060 REM  N = FIRST NUMBER ENTERED
065 REM  N1= SECOND NUMBER ENTERED
070 REM  I = LOOP CONTROL VARIABLE
072 REM  A = ANSWER ENTERED BY USER
075 REM  C = TOTAL OF CORRECT ANSWERS
080 REM  W = TOTAL OF INCORRECT ANSWERS
085 REM  C1= CORRECT ANSWER
087 RANDOMIZE
090 PRINT "THIS PROGRAM WILL PRINT 5 SETS OF NUMBERS."
091 PRINT "YOU WILL THEN ENTER THE ANSWERS FOR ADDITION
AND SUBTRACTION OF THE NUMBERS."
092 PRINT
99 FOR I = 1 TO 5
100 N = INT(41 * RND(0)) + 10
110 N1 = INT(41 * RND(0)) + 10
115 PRINT USING "## + ## = ",N,N1
120 INPUT A
122 C1 = N + N1
125 IF A = C1 THEN GOSUB 250:GOTO 140
130 GOSUB 300
140 PRINT USING "## - ## = ",N,N1
150 INPUT A1
155 C1 = N - N1
160 IF A1 = C1 THEN GOSUB 250:GOTO 180
170 GOSUB 300
180 NEXT I
200 PRINT USING "OUT OF 10 PROBLEMS, YOU ANSWERED ##
CORRECT AND ## INCORRECT",C,W
210 GOTO 999
250 REM SUBROUTINE FOR CORRECT ANSWERS
260 PRINT "YOUR ANSWER IS CORRECT"
270 C = C + 1
280 RETURN
300 REM SUBROUTINE FOR INCORRECT ANSWERS
310 PRINT " INCORRECT ANSWER "
320 PRINT "THE CORRECT ANSWER IS ";C1
330 W = W + 1
340 RETURN
999 END
```

5.

```
010 REM  THIS PROGRAM RANDOMLY SELECTS 5 WINNING RAFFLE
020 REM  TICKETS FROM NUMBERS 001 TO 500.
030 REM     I = LOOP CONTROL VARIABLE
```

```
090 RANDOMIZE
100 FOR I = 1 TO 5
110  PRINT "TICKET NUMBER";INT(RND(0)*501)
120 NEXT I
999 END
```

6.

```
010 REM THIS PROGRAM RANDOMLY SELECTS A NUMBER.  THE USER
020 REM IS THEN ALLOWED TEN GUESSES TO CHOOSE THE NUMBER.
030 REM THE COMPUTER WILL INDICATE IF THE NUMBER SELECTED
040 REM BY THE USER IS TOO HIGH OR TOO LOW.
050 REM     N = RANDOM NUMBER
060 REM     I = LOOP CONTROL VARIABLE
070 REM     G = GUESS
099 RANDOMIZE
100 N = INT(RND(0)*101)
105 FOR I = 1 TO 10
110 INPUT "I AM THINKING OF A NUMBER BETWEEN 1 AND 100 -
ENTER YOUR GUESS";G
120 IF G = N THEN PRINT "YOU GUESSED THE CORRECT
NUMBER":GOTO 999
130 IF G > N THEN PRINT "YOUR GUESS IS TOO HIGH - TRY
AGAIN"
140 IF G < N THEN PRINT "YOUR GUESS IS TOO LOW - TRY
AGAIN"
150 NEXT I
160 PRINT "YOU DIDN'T GUESS MY NUMBER IN 10 TRIES"
170 PRINT
180 PRINT "MY NUMBER WAS";N
999 END
```

7.

```
010 REM THIS PROGRAM ALLOWS THE USER TO INPUT 5 NUMBERS.
020 REM THE COMPUTER WILL THEN TELL THE USER IF THE NUMBER
030 REM IS EVEN OR ODD.
040 REM N = NUMBER ENTERED
050 REM I = LOOP CONTROL VARIABLE
100 FOR I = 1 TO 5
110   INPUT "ENTER A WHOLE NUMBER";N
120   IF N/2 = INT(N/2) THEN PRINT "THE NUMBER ";N;"IS
EVEN":GOTO 200
130   PRINT "THE NUMBER ";N;"IS ODD"
200 NEXT I
999 END
```

8.

```
010 REM THIS PROGRAM COMPUTES THE SALES TAX AND THE TOTAL
020 REM AMOUNT ON PURCHASES FOR THE STATES OF MISSISSIPPI,
030 REM ARKANSAS, TEXAS, AND CALIFORNIA.
040 REM
050 REM   A  = AMOUNT OF PURCHASE
055 REM   T  = TAX RATE
060 REM  S$ = STATE NAME
065 REM   S = AMOUNT OF SALES TAX
070 REM   P = TOTAL AMOUNT DUE
075 REM  A$ = ANSWER TO CONTINUE OR NOT CONTINUE PROGRAM
080 REM
100 INPUT "AMOUNT OF PURCHASE";A
```

```
110 T = .03
120 S$ = "MISSISSIPPI"
130 GOSUB 300
140 T = .04
150 S$ = "ARKANSAS"
160 GOSUB 300
170 T = .05
180 S$ = "TEXAS"
190 GOSUB 300
200 T = .06
210 S$ = "CALIFORNIA"
220 GOSUB 300
225 PRINT
230 INPUT "DO YOU WISH TO ENTER ANOTHER AMOUNT OF PURCHASE
- ENTER YES OR NO";A$
240 IF A$ = "YES" OR A$ = "Y" THEN GOTO 100
250 GOTO 999
300 REM  SUBROUTINE TO COMPUTE AND PRINT TAX AND TOTAL
305 REM  AMOUNT
310 S = T * A
320 P = S + A
325 PRINT
330 PRINT USING "   STATE    AMOUNT PURCHASED    SALES TAX
TOTAL AMOUNT DUE"
332 PRINT
335 PRINT USING "\           \  $#,###,###.##
$#,###,###.##    $#,###,###.##",S$,A,S,P
337 RETURN
340 IF A$ = "YES" OR A$ = "Y" THEN GOTO 100
350 GOTO 999
999 END
```

9.

```
010 REM THIS PROGRAM COMPUTES AND PRINTS PAYROLL INFORMATION
015 REM THE COMPANY PAYS TIME-AND-A-HALF FOR SATURDAY WORK,
020 REM DOUBLE-TIME FOR SUNDAY WORK, AND TIME-AND-A-HALF FOR
025 REM ALL HOURS OVER 40 ON MONDAY - FRIDAY.
030 REM
035 REM    N = = AMOUNT OF PAY FOR OVERTIME WORK
038 REM    F$ = FORMAT FIELD FOR PRINTOUT
040 REM    I  = LOOP CONTROL VARIABLE
042 REM    T  = TOTAL EARNINGS
044 REM    T1 = TOTAL HOURS MONDAY - FRIDAY
046 REM    H1 = TOTAL HOURS WORKED ALL WEEK
048 REM    O  = OVERTIME PAY
050 REM    N$ = NAME OF EMPLOYEE
052 REM    R  = RATE OF PAY
054 REM    H  = HOURS WORKED EACH DAY
056 REM    A  = AMOUNT EARNED FOR DAILY WORK
058 REM    D$ = DAY OF THE WEEK
060 REM
098 N = 0
099 F$ = "\           \        ##.##             $#,###.##"
100 FOR I = 1 TO 10
110   N = 0:T = 0: T1 = 0:H1 = 0
```

```
115   O = 0
120   READ N$,R
125   PRINT:PRINT:PRINT
130   PRINT USING "\                    \ $##.## PER HOUR",N$,R
140   REM  WAGES FOR SUNDAY
150   READ H
155   PRINT:PRINT
160   PRINT "     DAY           HOURS          AMOUNT
EARNED"
170   PRINT
175   A = H*R*2
180   D$ = "SUNDAY"
182   H1 = H1 + H
185   T = T + A
190   GOSUB 500
200   REM  WAGES FOR MONDAY - FRIDAY"
210   GOSUB 600
220   D$ = "MONDAY"
230   GOSUB 500
240   GOSUB 600
250   D$ = "TUESDAY"
260   GOSUB 500
270   GOSUB 600
280   D$ = "WEDNESDAY
290   GOSUB 500
300   GOSUB 600
310   D$ = "THURSDAY"
320   GOSUB 500
330   GOSUB 600
340   D$ = "FRIDAY"
350   GOSUB 500
361   REM COMPUTES SATURDAY PAT
362   READ H
363   A = H * R * 1.5
364   D$ = "SATURDAY"
365   T = T + A:H1=H1+H
366   GOSUB 500
370   IF T1 > 40 THEN N = T1 - 40:O = N*R*.5
380   T = T + O
390   D$ = "OVERTIME"
395   PRINT
400   PRINT USING F$,D$,N,O
405   PRINT
410   PRINT "TOTALS"
420   PRINT USING "HOURS WORKED        ###.##",H1
430   PRINT USING "TOTAL EARNINGS  $#,###.##",T
435   PRINT
440   PRINT "OVERTIME IS COMPUTED IN THE FOLLOWING MANNER"
445   PRINT
450   PRINT USING "TOTAL HOURS WORKED MONDAY - FRIDAY
##",T1
460   PRINT USING "OVERTIME ADDITIONAL PAY FOR ## HOURS
$#,###.##",N,O
470 NEXT I
```

```
480 GOTO 999
500 REM SUBROUTINE TO PRINT DATA
510 PRINT USING F$,D$,H,A
520 RETURN
600 REM SUBROUTINE FOR WAGES FOR MONDAY - FRIDAY
610 READ H
620 A = H * R
630 T = T + A
640 T1 = T1 + H
650 H1 = H1 + H
660 RETURN
800 DATA "DALE FINLEY",4,8,10,6,8,4,5,6
810 DATA "JUDY SWANSON",5,2,5,8,8,8,8,8
820 DATA "NATHAN HART",3.75,0,4,4,4,4,4,0
830 DATA "TROY MANDER",7.50,0,8,8,8,8,8,0
840 DATA "MARY FRANKLIN",8,2,8,8,10,8,8,0
850 DATA "ANTHONY STEWART",3.60,0,6,6,6,6,6,0
860 DATA "MATHEW DUMAS",10,0,8,8,8,8,4,4
870 DATA "KATHY MUNSON",12,1,8,10,10,8,8,5
880 DATA "MICHAEL JOHNSON",5,0,5,5,5,6,7,0
890 DATA "TIMOTHY RICHARDS",4.5,2,4,4,4,8,3,2
999 END
```

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

NAME______________________

TEST

1. Match the terms on the right with the correct definitions.

_____a. Automatic subroutines

_____b. Series of statements that constitute a program that can be executed only when "called" by another program

_____c. Returns the value of the greatest integer that is less than or equal to the argument

_____d. Returns the absolute value of the argument and determines how far the argument is from 0

_____e. Generates random numbers between 0 and 1, it displays a 6-digit fractional number

_____f. Returns the value of the sign only

_____g. Returns the square root of any positive value

_____h. BASIC statement used to transfer control back to the main program

_____i. An ordered set of numbers, each of which may not be predicted from any knowledge of its predecessors

_____j. A valid arithmetic expression enclosed in parentheses that is used with BASIC functions

_____k. Subroutines stored in the library of the computer's memory that are used in programs when special operations are to be performed

_____l. A data item used to begin a series of calculations

_____m. BASIC statement used to transfer control to subroutines

1. ABS function
2. Argument
3. Built-in functions
4. Function
5. GOSUB statement
6. INT function
7. Random sequence
8. RETURN statement
9. RND function
10. Seed
11. SGN function
12. SQR function
13. Subroutine

2. Select true statements concerning built-in functions by placing an "X" in the appropriate blanks.

_____a. The SQR function will print an error message if the value of the argument is negative

_____b. Decimal numbers cannot be randomly generated on most computers

_____c. The return of the value of INT(3.44) is 3.44

_____d. The statement RANDOM or RANDOMIZE ensures that an unpredictable sequence of random numbers will be generated each time the program is executed

_____e. Arguments can be single variables, single constants, or any valid mathematical expression

_____f. RND function is useful in games

_____g. Three unique identifying letters are used with built-in functions

_____h. The ABS function determines how far the argument is from 0

_____i. The statement X = INT(60 * RND - 30) or X = INT (60 * RND(0) - 30) produces random numbers from 30 to 60

_____j. When the value of the argument in the SGN function is 0, the value of the sign is -

_____k. Built-in functions are subroutines that are stored in the computer's memory

_____l. Six-digit fractional numbers are generated when the RND(0) is used on the TRS-80 or RND is used on the PDP 11/34 computer

_____m. The ABS function returns the argument as a positive value only

_____n. To round off numbers using the INT function, .5 is added to the argument

_____o. On the TRS-80, RND(10) generates positive integer values up to 10

_____p. The return of the value of INT (-3.44) is -3

_____q. Random numbers can be generated in a range of values _____r. The SGN function returns the value of the argument and the + or - sign

_____s. The RND function generates random numbers from 0 to 1

_____t. The RND function operates the same on all computers

3. Select true statements concerning subroutines by placing an "X" in the appropriate blanks.

_____a. A GOSUB statement can be included in a program without a RETURN statement

_____b. More than one subroutine may be used in a program

_____c. Control is returned to the line following the GOSUB program at the end of a subroutine

_____d. Subroutines are "called" by a GOSUB statement in the main program

_____e. A subroutine can "call" itself

_____f. Control is transferred back and forth from the main program and the subroutine

_____g. The line numbers of a subroutine can be the same as those in the main program

_____h. Only one RETURN can be used in each subroutine

_____i. A RETURN statement or an END statement marks the end of a subroutine

_____j. The first line of a subroutine should be a REM statement with an explanation of what the subroutine does

_____k. Control can be transferred into a subroutine through a GOTO statement or a GOSUB statement

_____l. Any valid BASIC statement can be used in a subroutine _____m. The following is a valid subroutine:

```
100 GOSUB 200
110 GOSUB 300
120 GOSUB 400
200 REM PRINTS FIRST LINE
210 PRINT "LINE ONE"
220 RETURN
300 REM PRINTS SECOND LINE
310 PRINT "LINE TWO"
400 PRINT "LINE THREE"
500 RETURN
999 END
```

_____n. A subroutine can "call" another subroutine

_____o. The GOSUB statement causes the computer to store an address to return to after the subroutine is completed

4. Write the results of selected built-in functions.

5. Determine the output of programs containing subroutines.

6. Flowchart, code, enter, and RUN programs using built-in functions and subroutines.

(NOTE: If activities 4 - 6 have not been accomplished prior to the test, ask your instructor when they should be completed.)

BUILT-IN FUNCTIONS AND SUBROUTINES
UNIT IX

ANSWERS TO TEST

1. a. 4
   b. 13
   c. 6
   d. 1
   e. 9
   f. 11
   g. 12
   h. 8
   i. 7
   j. 2
   k. 3
   l. 10
   m. 5

2. a.
   d.
   e.
   f.
   g.
   h.
   k.
   l.
   m.
   n.
   o.
   q.

3. b.
   c.
   d.
   f.
   j.
   l.
   n.
   o.

4. Evaluated to the satisfaction of the instructor.

5. Evaluated to the satisfaction of the instructor.

6. Evaluated to the satisfaction of the instructor.

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS

UNIT X

UNIT OBJECTIVE

After completion of this unit, the student should be able to select true statements associated with subscripted variables, the DIMension statement, and single- and double-dimensional arrays; select true statements concerning subscripted variables, single-dimensional arrays, and the DIM statement; write FOR/NEXT loops to load a single-dimensional array, manipulate data in the array, and print the entire array; and select true statements concerning double- dimensional arrays. The student should also be able to identify elements in single- and double-dimensional arrays by location and values; write the results of programs containing single- and double-dimensional arrays; write statements to dimension, load, manipulate, and process single- and double- dimensional arrays; and flowchart, code, enter, and RUN programs containing single- and double-dimensional arrays. This knowledge will be evidenced by correctly performing the procedures outlined in the assignment sheets and by scoring 85 percent on the unit test.

SPECIFIC OBJECTIVES

After completion of this unit, the student should be able to:

1. Match terms associated with subscripted variables, the DIMension statement, and single- and double-dimensional arrays.
2. Select true statements concerning subscripted variables and single-dimensional arrays.
3. Select true statements concerning the DIMension statement.
4. Write FOR/NEXT loops to load a single-dimensional array, manipulate data in the array, and print the entire array.
5. Select true statements concerning double-dimensional arrays.
6. Identify elements in single- and double-dimensional arrays by locations and values.
7. Write the results of programs containing single- and double-dimensional arrays.

8. Write statements to dimension, load, manipulate, and process single- and double-dimensional arrays.

9. Flowchart, code, enter, and RUN programs containing single- and double-dimensional arrays.

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

SUGGESTED ACTIVITIES

I. Provide student with objective sheet.

II. Provide student with information and assignment sheets.

III. Discuss unit and specific objectives.

IV. Discuss information and assignment sheets.

V. Give test.

INSTRUCTIONAL MATERIALS

I. Included in this unit:

  A. Objective sheet

  B. Information sheet

  C. Assignment sheets

    1. Assignment Sheet #1--Identify elements in single- and double-dimensional arrays by location and values

    2. Assignment Sheet #2--Write the results of programs containing single- and double-dimensional arrays

    3. Assignment Sheet #3--Write statements to dimension, load, manipulate, and process single- and double-dimensional arrays

    4. Assignment Sheet #4--Flowchart, code, enter, and RUN programs containing single- and double-dimensional arrays

  D. Answers to assignment sheets

  E. Test

  F. Answers to test

II. References:

Refer to Unit I references.

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

INFORMATION SHEET

I. Terms and definitions

  A. Subscripted variables - variables that store related numeric and string values that can be referenced with one variable name having single or double subscripts

  B. Subscript - the number or variable that shows the location of one element in an array (enclosed in parentheses right after the array name)

  C. Array - a list of related numeric or string values that are of the same type and use the same variable name

  D. Element - a single position in an array (each is equivalent to a variable)

  E. Single-dimension array - table with values in columnar locations only (list)

  F. Dimension - to declare the name and size of an array to the computer

  G. DIM statement - BASIC statement that is used to reserve space in the computer's memory for an array

  H. Double-dimension array - table with values in locations of rows and columns

  I. Row - the first subscript in an array, usually represented as moving across a page (horizontal)

  J. Column - a second subscript of an array, usually represented as moving down a page (vertical)

II. Subscripted variables and single-dimensional arrays

  A. The purpose of subscripted variables and arrays is to provide a fast and orderly way of storing and retrieving large quantities of data

  B. Types of subscripted variables

    1. Any valid numeric and string variable can be subscripted

INFORMATION SHEET

2. Numeric

   Examples:

   A(15)
   B1(20)

3. String

   Example:

   N$(50)

C. The subscript is the number in parentheses used to refer to the location of the element in storage

D. The list or table of the subscripted variables is referred to as an array

E. On most systems, arrays begin with element 0

(NOTE: The zero subscript is often not used and will not be used in this unit.)

F. A subscripted variable is referred to as - array name sub-position (element)

Example:

A(10)	A sub-10	The tenth element in array A
B(6)	B sub-6	The sixth element in array B
C$(8)	C$ sub-8	The eighth element in array C$

G. An element in an array can be referenced in three ways

1. Using a constant A(3)
2. Using a variable A(N)
3. Using an arithmetic expression A(N*2)

   (NOTE: On some computers, e.g., IBM S/34, BASIC will not permit use of the same name for subscripted and unsubscripted variables.)

H. A variable name can be used both as a subscripted variable and as an unscripted variable in the same program

INFORMATION SHEET

Example:

```
10 A = 16
20 A(1) = 24
```

I. Each element in an array can be used exactly as a single variable is used

III. DIM (dimension) statement

A. The DIMension statement is used to reserve memory space in the computer to store an array

Example:

```
100 DIM A(10)
110 DIM B$(10)
```

(NOTE: The statements above tell the computer to set aside 10 spaces in memory for the array A and 10 spaces for the array B$.)

B. General format

```
                               100 DIM A(40)
                                 |   |  |  |
1.  Line number__________________|   |  |  |
                                     |  |  |
2.  The word DIM_____________________|  |  |
                                        |  |
3.  The name of the array_______________|  |
                                           |
4.  Positive integer value_________________|
    specifying the maximum
    size of the array
```

C. DIM statements can be placed anywhere in the program

(NOTE: It is a good practice to place the DIM statement at the beginning of the program so it can be changed easily when necessary.)

D. On most computers, the maximum size in the DIM statement must be a specific positive integer value

Example:

100 DIM N(B) would be invalid

INFORMATION SHEET

E. A DIM statement is not required for arrays of 11 or under elements

A(0) - A(10)

(NOTE: It is a good programming practice to DIM all arrays.)

F. The same variable cannot be dimensioned more than once

G. More than one array may be dimensioned at once

Example:

```
100 DIM A(10), B(6), A$(25)
```

H. If DIM size is smaller than number of elements used in the array, an error message will be printed

Example:

```
100 DIM A(8)
110 READ A(9)
```

```
SUBSCRIPT OUT OF RANGE AT LINE 110
```

I. The maximum size of the array should be for more elements than are actually needed

IV. Processing data in an array

A. Create the array with DIM statement

B. Storing values in an array (loading an array)

1. Values can be stored in an array in the same way values are stored in variables

a. READ statement

```
10 READ A(1), A(2), A(3), A(4), A(5)
20 DATA 2,4,6,8,10
```

b. LET statement

```
10 LET A(1) = 2
20 LET A(2) = 4
30 LET A(3) = 6
40 LET A(4) = 8
50 LET A(5) = 10
```

INFORMATION SHEET

c. INPUT statement

```
10 INPUT A(1)
20 INPUT A(2)
30 INPUT A(3)
40 INPUT A(4)
50 INPUT A(5)
```

(NOTE: Values 2,4,6,8,10 would be entered during RUN of the program.)

2. Most used and convenient way is through the FOR/NEXT loop

Example:

```
10 FOR I = 1 TO 5
20   READ A(I)
30 NEXT I
40 DATA 2,4,6,8,10
```

(NOTE: Any variable can be used in place of the variable I.)

3. The values in the above examples produce the following array A

		Value
Location	A(1)	2
	A(2)	4
	A(3)	6
	A(4)	8
	A(5)	10

(NOTE: In the above example, the value of A sub-1 is 2, the value 10 is in the 5th position of the array A.)

C. Manipulating data in an array

1. Searching an array

Example:

```
.
.
200 FOR X = 1 TO 10
210   IF Z(X) = 1472 THEN 300
220 NEXT X
230 PRINT "ITEM NOT FOUND"
240 GOTO 999
300 PRINT "ITEM FOUND"
999 END
```

INFORMATION SHEET

2. Calculating values in an array
    a. Elements in an array can be used in mathematical calculations
    b. One or more elements may be used at a time

        Example:

```
100 T2 = X(1) + A * B
110 FOR I = 2 TO 15
120   T = T + X(I)
130 NEXT I
```

3. Printing the values of an array
    a. An array can be printed in the same way variables are printed
    b. Elements can be printed one or more at a time
    c. Most convenient way to print entire array is to use a FOR/NEXT loop

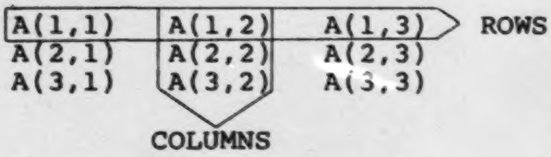
        Example:

```
100 FOR J = 1 TO 50
110   PRINT X(J)
120 NEXT J
```

V. Double-dimensional arrays

A. Two subscripts are used to identify elements in a double-dimensional array
    1. First subscript represents the row
    2. Second subscript represents the column

        Example:



A(1,1)	A(1,2)	A(1,3)
A(2,1)	A(2,2)	A(2,3)
A(3,1)	A(3,2)	A(3,3)

INFORMATION SHEET

B. General format

```
         100 X$(6,7)
or       100 X(2,3)
or       100 X(R,C)
```

a. Line number

b. The name of the array

c. The row position

d. The column position

(NOTE: The variables R and C are used only for convenience in the program. Any valid numeric variable name, constant, or arithmetic expression may be used.)

C. A DIM statement is required for double-dimensional arrays

1. General format

```
100 DIM X(4,5)
```

a. Line number

b. The word DIM

c. The name of the array

d. The specific positive integer value indicating the maximum size of the row position

e. The specific positive integer value indicating the maximum size of the column position

(NOTE: The above example would set aside space in the memory of the computer for an array of 4 rows and 5 columns.)

INFORMATION SHEET

Example:

Columns

Rows	1	2	3	4	5
1					
2					
3					
4					

2. The row and column maximum sizes must be specific positive integer values in the DIM statement

D. A nested loop can be used to load and manipulate a double-dimensional array

1. Loading values into array

```
100 DIM X(3,4)
110 FOR R = 1 TO 3
120   FOR C = 1 TO 4
130     READ X(R,C)
140   NEXT C
150 NEXT R
800 DATA 2,4,6,8,10,12,14,16,18,20,22,24
999 END
```

COLUMNS

ROWS	C= 1	C= 2	C= 3	C= 4	
R = 1	(1,1) 2	(1,2) 4	(1,3) 6	(1,4) 8	Location Value
R = 2	(2,1) 10	(2,2) 12	(2,3) 14	(2,4) 16	Location Value
R = 3	(3,1) 18	(3,2) 20	(3,3) 22	(3,4) 24	Location Value

(NOTE: Use the above illustration for the programs on vertical and horizontal manipulation below.)

2. Two-dimensional arrays can be manipulated in two ways

INFORMATION SHEET

a. Horizontally

```
100 DIM X(3,4)
110 FOR R = 1 TO 3
120   FOR C = 1 TO 4
125 READ X(R,C)
130     PRINT X(R,C);
140   NEXT C
150 NEXT R
800 DATA 2,4,6,8,10,12,14,16,18,20,22,24
999 END

RUN

 2  4  6  8  10  12  14  16  18  20  22  24
```

b. Vertically

```
100 DIM X(3,4)
110 FOR C = 1 TO 4
120   FOR R = 1 TO 3
125 READ X(R,C)
130     PRINT X(R,C);
140   NEXT R
150 NEXT C
800 DATA 2,4,6,8,10,12,14,16,18,20,22,24
999 END

RUN

 2  10  18  4  12  20  6  14  22  8  16  24
```

E. Same variable cannot be used as a single- and double-subscripted variable

Example:

A(20) and A(14,15) cannot be used in same program

A and A(14,15) can be used in same program

F. Same variable can be used as an unsubscripted variable and as a double-subscripted variable

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

ASSIGNMENT SHEET #1--Identify elements in single- and double-dimensional arrays by locations and values

Driections: Using the following tables, write the values or locations of the elements given in the single- or double-dimensional arrays given. Each array begins with element 1 instead of 0.

1. Write the values of array A for the locations listed below:

ARRAY A
2
6
9
3
5

a. ________A(3)

b. ________A(5)

c. ________A(1)

2. Write the location of the following values in array B:

ARRAY B

4
6
8
10

a. ________6

b. _______10

c. _______ 4

ASSIGNMENT SHEET #1

3. Write the values for array C$ listed below:

ARRAY C$

ADAMS
BATES
CARTER
DODGE
EAST

a. __________ C$(2)

b. __________ C$(5)

c. __________ C$(3)

4. Write the location in array D$ of the following strings:

ARRAY D$

HODGES
INMAN
JONES
KING
LOFTON

a. __________ LOFTON

b. __________ KING

c. __________ HODGES

5. Write the values for the following double-dimensional array E:

ARRAY E

2	4	6	8	10
12	14	16	18	20
22	24	26	28	30

a. __________ E(2,2)

d. __________ E(2,1)

b. __________ E(3,4)

e. __________ E(2,5)

c. __________ E(1,2)

ASSIGNMENT SHEET #1

6. Write the location of the following values in the double-dimensional array F:

ARRAY F

12	16	5	3	8
30	36	6	99	11
4	80	10	40	1

a. __________ 16

b. __________ 4

c. __________ 11

d. __________ 40

e. __________ 30

7. Write the value of the following locations in the double-dimensional array G$:

ARRAY G$

DOG	CAT	MONKEY	ELEPHANT
LION	ZEBRA	TURTLE	GOAT
BEAR	SNAKE	DUCK	FISH

a. __________ BEAR

b. __________ DUCK

c. __________ CAT

d. __________ LION

e. __________ FISH

8. Write the location of the following values stored in array H$:

ARRAY H$

ARRAY	BASIC	DIM	NESTED
STATEMENT	REM	PRINT	FOR
NEXT	READ	DATA	IF/THEN
GOTO	INPUT	INT(10)	SUBROUTINE

a. __________ H$(3,1)

b. __________ H$(1,2)

c. __________ H$(4,3)

d. __________ H$(2,1)

e. __________ H$(2,4)

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

ASSIGNMENT SHEET #2--Write the results of programs containing single- and double-dimensional arrays

Directions: For the programs listed below, write the result that would be printed.

1.
```
100 DIM A(20)
200 FOR J = 1 TO 20
210   A(J) = J * 2
220 NEXT J
230 PRINT A(3);A(10)
999 END
```

____

2.
```
100 DIM A(6)
200 FOR J = 1 TO 6
210   A(J) = J
220 NEXT J
230 FOR I = 1 TO 6
240   PRINT A(I)
250 NEXT I
999 END
```

____

3.
```
100 DIM B(6)
200 FOR L = 1 TO 6
210   B(L) = L
220 NEXT L
230 FOR K = 1 TO 4
240   B(K) = B(K+1)
250 NEXT K
260 FOR M = 1 TO 5
270   PRINT B(M);
280 NEXT M
999 END
```

____

ASSIGNMENT SHEET #2

4.
```
100 DIM C(100)
200 FOR L = 1 TO 5
210   C(L) = 0
220 NEXT L
230 FOR M = 6 TO 10
240   C(M) = 1
250 NEXT M
260 FOR N = 11 TO 15
270   C(N) = 2
280 NEXT N
290 FOR P = 1 TO 15
300   PRINT C(P);
310 NEXT P
999 END
```

____

5.
```
100 DIM A$(3),B(3),C(3)
200 C = 3
210 FOR I = 1 TO C
220   READ A$(I),B(I)
230 NEXT I
240 FOR J = 1 TO C
250   C(J) = 2 * B(J)
260 NEXT J
270 FOR K = 1 TO C
280   PRINT A$(K), B(K), C(K)
290 NEXT K
800 DATA "MIKE",2,"MARTY",4,"MARILYN",6
999 END
```

____

____

____

6.
```
100 DIM A(3,4)
110 FOR R = 1 TO 3
115   PRINT
120   FOR C = 1 TO 4
130     READ A(R,C)
135     PRINT A(R,C);
140   NEXT C
150 NEXT R
800 DATA 1,2,3,4,2,4,6,8,3,5,7,9
999 END
```

____

____

____

ASSIGNMENT SHEET #2

7.
```
100 DIM A$(2,4)
200 FOR R = 1 TO 2
210   FOR C = 1 TO 4
220     READ A$(R,C)
230   NEXT C
240 NEXT R
300 FOR R = 1 TO 2
310   PRINT
320   FOR C = 1 TO 4
330     PRINT A$(R,C),
340   NEXT C
350 NEXT R
800 DATA "ONE","TWO","THREE","FOUR","FIVE","SIX","SEVEN","EIGHT"
999 END
```

____

____

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

ASSIGNMENT SHEET #3--Write statements to dimension, load, manipulate, and print single- and double-dimensional arrays

Directions: Code statements to perform each of the following processes:

1. Code a DIM statement and a FOR/NEXT loop to enter the following data into the single-dimensional array A.

   DATA: 100,200,150,300,600

   ____

   ____

   ____

   ____

   ____

   ____

2. Code a FOR/NEXT loop to print the values stored in array A of program 1.

   ____

   ____

   ____

3. Code a FOR/NEXT loop to add the elements of array A and to assign the total of the elements to the variable S. Print the totals.

   ____

   ____

   ____

4. Code a DIM statement and a nested FOR/NEXT loop to enter the following data into the double-dimensional array D of 3 rows and 4 columns.

   DATA: 45,67,89,12,34,56,78,1,2,3,4,5,6

ASSIGNMENT SHEET #3

5. Code a nested FOR/NEXT loop to add the values of each row in array D. Assign the sum of row 1 to R(1), sum of row 2 to R(2), and the sum of row 3 to R(3).

6. Code a nested FOR/NEXT loop to print the values of array D in 3 rows and 4 columns and the value of R(1), R(2), and R(3) on 3 separate lines.

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

ASSIGNMENT SHEET #4--Flowchart, code, enter, and RUN programs containing single- and double-dimensional arrays

1. Flowchart, code, enter, and RUN a program to store and print the following values into two arrays:

   DATA: 2,3,5,6,1,3,4,5,6,7

   DATA: JOHN,MARY,SUE,BILL,PETER,AGNES,LUCY,CHARLIE

2. Flowchart, code, enter, and RUN a program to store and print a table that will appear as follows:

```
10          30             90             50
60          40             60             20
```

3. Print the above data in the following manner:

```
10            60
30            40
90            60
50            20
```

4. Flowchart, code, enter, and RUN a program to print the multiplication tables from 0 to 12.

5. Flowchart, code, enter, and RUN a program that will READ the months of the year into a single-dimensional array. Use an INPUT statement to enter a number for 1 to 12. The program will then have the month that corresponds to the numeric value printed on the terminal.

6. Flowchart, code, enter, and RUN a program that will create a table of 10 stock numbers and cost. The program will then allow a stock number to be input from the terminal, a search be made to locate the stock number, and the item number and cost be printed when it is located. If the item is not located, a message will be printed indicating the item was not found and instruct the user to enter another number in order to continue.

ASSIGNMENT SHEET #4

Stock Number	Cost
1000	24
1010	40
1020	50
1030	10
1040	35.66
1050	24.55
1060	45
1070	33.44
1080	228.22
1090	222

7. Flowchart, code, enter, and RUN a program that will read in and store the data listed below in a double-dimensional array:

Month	Department					
	1	2	3	4	5	6
1	20000	4000	900	40000	10000	6000
2	15000	2000	200	20000	8000	4000
3	30000	6000	500	20000	7000	3000
4	40000	6000	800	40000	10000	4000
5	20000	5000	600	30000	6000	5000
6	15000	1000	700	15000	10000	6000

Print out the following:

a. The total array
b. The amounts for the first 3 months for all departments
c. The amounts for the last 3 months for all departments
d. The amounts for months 1, 3, and 5 for all departments
e. The amounts for months 2, 4, and 6 for all departments
f. The amounts for departments 1 and 3 (all months)
g. The amounts for departments 2, 3, 4, 5 (all months)
h. The amounts for departments 4, 5, 6 (all months)
i. The amounts for departments 1 and 4 (all months)
j. The amounts for department 3 (all months)

Calculate the total and print for:

a. Each month
b. Each three months
c. The entire six months
d. Each department (all months)

ASSIGNMENT SHEET #4

8. The prices of four cars are listed below. The number of cars sold by each salesperson is also shown.

Flowchart, code, enter, and RUN a program using the following data:

Model No.	Price
1	8,000
2	12,000
3	15,000
4	18,000

Salesperson	Cars Sold by Models			
	1	2	3	4
Jones	2	6	4	8
Thomas	4	8	8	6
Carlton	8	9	6	4
Bates	3	6	2	3
Hughes	5	2	1	1

Calculate and print the following:

a. Total dollar sales for each sales representative
b. Total dollar sales for all sales representatives
c. Total number of each model sold
d. Total number of all cars sold
e. The percent of the total cars sold by models

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

SUGGESTED ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet #1

1. a. 9
   b. 5
   c. 2

2. a. B(2)
   b. B(4)
   c. B(1)

3. a. BATES
   b. EAST
   c. CARTER

4. a. D$(5)
   b. D$(4)
   c. D$(1)

5. a. 14
   b. 28
   c. 4
   d. 12
   e. 20

6. a. F(1,2)
   b. F(3,1)
   c. F(2,5)
   d. F(3,4)

7. a. G$(3,1)
   b. G$(3,3)
   c. G$(1,2)
   d. G$(2,1)
   e. G$(3,4)

8. a. NEXT
   b. BASIC
   c. INT(10)
   d. STATEMENT
   e. FOR

Assignment Sheet #2

```
1.  6  20

2.  1  2  3  4  5  6

3.  2  3  4  5  5

4.  0  0  0  0  0  1  1  1  1  1  2  2  2  2  2

5.  MIKE           2           4
    MARTY          4           8
    MARILYN        6          12

6.  1  2  3  4
    2  4  6  8
    3  5  7  9

7.  ONE            TWO           THREE          FOUR
    FIVE           SIX           SEVEN          EIGHT
```

Assignment Sheet #3

1.
```
100 DIM A(5)
110 FOR I = 1 TO 5
120   READ A(I)
130 NEXT I
800 DATA 100,200,150,300,600
```

2.
```
200 FOR I = 1 TO 5
210   PRINT A(I)
220 NEXT I
```

3.
```
200 FOR I = 1 TO 5
210  S = S + A(I)
220 NEXT I
230 PRINT S
```

4.
```
100 DIM D(3,4)
110 FOR R = 1 TO 3
120   FOR C = 1 TO 4
130     READ D(R,C)
140   NEXT C
150 NEXT R
800 DATA 45,67,89,12,34,56,78,1,2,3,4,5,6
999 END
```

5.
```
100 DIM D(3,4), R(3)
110 FOR R = 1 TO 3
120   FOR C = 1 TO 4
130     READ D(R,C)
140   NEXT C
150 NEXT R
200 FOR R = 1 TO 3
210   FOR C = 1 TO 4
220     R(I) = R(I) + D(R,C)
230   NEXT C
240 NEXT R
999 END
```

6.
```
100 DIM D(3,4), R(3)
105 LET F$ = "###   ###   ###   ###"
106 LET F1$ = "THE TOTAL OF ROW # IS ####"
110 FOR R = 1 TO 3
120   FOR C = 1 TO 4
130     READ D(R,C)
140   NEXT C
150 NEXT R
200 FOR R = 1 TO 3
205  X(R) = 0
210   FOR C = 1 TO 4
220     X(R) = X(R) + D(R,C)
230   NEXT C
240 NEXT R
300 FOR R = 1 TO 3
```

```
310   FOR C = 1 TO 4
320      PRINT USING F$,D(R,C);
330   NEXT C
335 PRINT
340 NEXT R
345 PRINT
350 FOR R = 1 TO 3
360   PRINT USING F1$,R,X(R)
370 NEXT R
800 DATA 45,67,89,12,34,56,78,1,2,3,4,5,6
999 END
```

Assignment Sheet #4

1.

```
100 REM  THIS PROGRAM STORES AND PRINTS TWO
110 REM  SINGLE-DIMENSIONAL ARRAYS
120 REM     I AND J = LOOP CONTROL VARIABLES
130 REM       A(10) = NUMERIC ARRAY
140 REM      N$(10) = STRING ARRAY
150 DIM A(10),N$(10)
160 FOR I = 1 TO 10
170   READ A(I)
180 NEXT I
190 FOR J = 1 TO 10
200   READ N$(J)
210 NEXT J
220 FOR I = 1 TO 10
230   PRINT A(I);
240 NEXT I
250 PRINT
260 FOR J = 1 TO 10
270   PRINT N$(J)
280 NEXT J
300 DATA 2,3,5,6,1,3,4,5,6,7
310 DATA"JOHN","MARY","SUE","BILL","PETER","AGNES",
"LUCY","CHARLIE"
320 DATA "POWELL","ANTHONY"
999 END
```

2.

```
100 REM  THIS PROGRAM STORES AND PRINTS A
110 REM  DOUBLE-DIMENSIONAL ARRAY
120 REM       R = ROW
130 REM       C = COLUMN
140 REM  A(R,C) = DOUBLE-DIMENSIONAL ARRAY
150 DIM A(2,4)
160 FOR R = 1 TO 2
170   FOR C = 1 TO 4
180     READ A(R,C)
190   NEXT C
200 NEXT R
210 FOR R = 1 TO 2
220   FOR C = 1 TO 4
230     PRINT USING "###   ###   ###   ###",A(R,C);
```

```
240 NEXT C
250 PRINT
260 NEXT R
300 DATA 10,30,90,50,60,40,60,20
999 END
```

3.
```
100 REM  THIS PROGRAM PRINTS THE DOUBLE-DIMENSIONAL ARRAY
101 REM  IN PROGRAM 1 IN A DIFFERENT FORMAT
105 DIM A(2,4)
110 FOR R = 1 TO 2
120   FOR C = 1 TO 4
130     READ A(R,C)
140   NEXT C
150 NEXT R
200 FOR C = 1 TO 4
210   FOR R = 1 TO 2
220     PRINT USING "###         ###",A(R,C);
230 NEXT R
235 PRINT
240 NEXT C
800 DATA 10,30,90,50,60,40,60,20
999 END
```

4.
```
100 REM  THIS PROGRAM PRINTS THE MULTIPLICATION TABLES
110 REM  FROM 0 TO 12
120 REM       R = ROW
130 REM       C = COLUMN
140 REM  A(R,C) = ARRAY FOR TOTALS OF THE TABLES
150 DIM A(12,12)
160 FOR R = 0 TO 12
170   FOR C = 0 TO 12
180     A(R,C) = R* C
190   NEXT C
200 NEXT R
210 FOR R = 0 TO 12
220   PRINT
230   FOR C = 0 TO 12
240     PRINT USING "## X ## = ###",R,C,A(R,C)
250   NEXT C
260 NEXT R
999 END
```

5.
```
100 REM  THIS PROGRAM ALLOWS THE USER TO INPUT A NUMBER
110 REM  FROM 1 TO 12.  THE COMPUTER THEN PRINTS THE
120 REM  CORRESPONDING MONTH OF THE NUMBER
130 REM      M$(12) = STRING ARRAY OF MONTHS
140 REM           M = NUMERIC VALUE OF MONTH ENTERED
150 REM           I = LOOP CONTROL VARIABLE
160 DIM M$(12)
170 INPUT "ENTER A NUMBER FROM 1 - 12 TO REPRESENT THE
MONTH DESIRED",M
180 FOR I = 1 TO 12
190   READ M$(I)
200 NEXT I
```

```
    210 PRINT USING "MONTH ## IS \                \",M,M$(M)
    800 DATA"JANUARY","FEBRUARY","MARCH","APRIL","MAY",
    810 DATA "JUNE","JULY","AUGUST","SEPTEMBER","OCTOBER",
    820 DATA"NOVEMBER","DECEMBER"
    999 END
```

```
6.  100 REM  THIS PROGRAM CREATES AN ARRAY OF ITEM NUMBER AND
    110 REM  COSTS.  THE USER ENTERS AN ITEM NUMBER AND THE
    120 REM  COST WILL BE PRINTED.
    130 REM     S(10,2) = ARRAY WITH ITEM NUMBERS AND COSTS
    140 REM           S = ITEM NUMBER ENTERED TO BE LOCATED
    150 REM           R = ROW OF THE ARRAY
    160 REM           C = COLUMN OF ARRAY
    170 REM           A = LOCATION OF DESIRED ITEM NUMBER
    180 DIM S(10,2)
    190 FOR R = 1 TO 10
    200   FOR C = 1 TO 2
    210   READ S(R,C)
    220   NEXT C
    230 NEXT R
    240 INPUT "ENTER THE STOCK NUMBER OF THE ITEM DESIRED";S
    250 FOR R = 1 TO 10
    260   IF S(R,1) = S THEN A = R:GOTO 300
    270 NEXT R
    280 PRINT USING "ITEM #### NOT FOUND",S
    290 GO TO 999
    300 PRINT USING "ITEM #### COST $$####.##",S(A,1),S(A,2)
    800 DATA 1000,24,1010,40,1020,50,1030,10,1040,35.66,
    810 DATA 1050,24.55,1060,45,1070,33.44,1080,228.22,1090,222
    999 END
```

```
7.  002 REM  THIS PROGRAM IS USED TO STORE, MANIPULATE, AND
    004 REM  PRINT DATA IN A DOUBLE-DIMENSIONAL ARRAY.
    006 REM  THE ARRAY CONSISTS FOR 6 MONTHS (ROWS) AND 6
    008 REM  DEPARTMENTS (COLUMNS).
    010 REM  THE FOLLOWING ITEMS ARE PRINTED:
    012 REM
    014 REM   1.  THE TOTAL ARRAY
    016 REM   2.  THE AMOUNTS FOR THE FIRST 3 MONTHS FOR ALL
    018 REM       DEPARTMENTS
    020 REM   3.  THE AMOUNTS FOR THE LAST 3 MONTHS FOR ALL
    022 REM       DEPARTMENTS
    024 REM   4.  THE AMOUNTS FOR MONTHS 1, 3, AND 5 FOR ALL
    026 REM       DEPARTMENTS
    028 REM   5.  THE AMOUNTS FOR MONTHS 2, 4, AND 6 FOR ALL
    030 REM       DEPARTMENTS
    032 REM   6.  THE AMOUNTS FOR DEPARTMENTS 1 AND 3 FOR ALL
    034 REM       MONTHS
    036 REM   7.  THE AMOUNTS FOR DEPARTMENTS 2, 3, 4, AND 5
    038 REM       FOR ALL MONTHS
    040 REM   8.  THE AMOUNTS FOR DEPARTMENTS 4, 5, AND 6 FOR
    042 REM       ALL MONTHS
    044 REM   9.  THE AMOUNTS FOR DEPARTMENTS 1 AND 4 FOR ALL
    046 REM       MONTHS
```

```
048 REM  10.  THE AMOUNTS FOR DEPARTMENT 3 FOR ALL MONTHS
050 REM  11.  THE TOTALS FOR EACH MONTH
052 REM  12.  THE TOTALS FOR EACH THREE MONTHS
054 REM  13.  THE TOTALS FOR ALL 6 MONTHS
056 REM  14.  THE TOTAL FOR EACH DEPARTMENT FOR ALL MONTHS
058 REM
060 REM       A(6,6) = ARRAY FOR MONTHS AND DEPARTMENTS
062 REM       R      = ROW OF ARRAY
064 REM       C      = COLUMN OF ARRAY
066 REM       B      = BEGINNING MONTH
068 REM       M      = ENDING MONTH
070 REM       S      = STEP FOR MONTHS
072 REM       B1     = BEGINNING DEPARTMENT
074 REM       D      = ENDING DEPARTMENT
076 REM       S1     = STEP FOR DEPARTMENTS
078 REM       F$,F1$ = FORMAT FIELDS
080 REM       T      = TOTAL FOR FIRST THREE MONTHS
082 REM       T(6)   = ARRAY FOR MONTHLY TOTALS
084 REM       T1     = TOTAL FOR LAST 3 MONTHS
086 REM       T2     = TOTAL FOR ALL 6 MONTHS
088 REM       T1(6)  = ARRAY FOR DEPARTMENTAL TOTALS
090 REM       H      = VARIABLE USED TO PRINT DEPT. NO.
092 REM                HEADINGS
094 REM
100 DIM A(6,6),T(6), T1(6)
110 F$ = "    ######  ######  ######  ######  ######
######"
120 F1$ = "##"
130 FOR R = 1 TO 6
140   FOR C = 1 TO 6
150     READ A(R,C)
160   NEXT C
170 NEXT R
180 B = 1:M = 6:S = 1
190 B1 = 1:D = 6:S1 = 1
200 GOSUB 702
210 B = 1:M = 3:S = 1
220 B1 = 1:D = 6:S1 = 1
230 GOSUB 702
240 B = 4:M = 6
250 GOSUB 702
260 B = 1:M = 5:S = 2
270 GOSUB 702
280 B = 2:M = 6:S = 2
290 GOSUB 702
300 D = 3:S1 = 2
310 B = 1:M = 6: S = 1
320 GOSUB 702
330 B1 = 2:D = 5:S1 = 1
340 GOSUB 702
350 B1 = 4:D= 6
360 GOSUB 702
370 B = 1:M = 6:S = 1
380 B1 = 1:D = 4:S1 = 3
```

```
390 GOSUB 702
400 B1 = 3:D = 3:S = 1
410 GOSUB 702
420 B = 1:M=6:S = 1
430 B1 = 1:D = 6: S = 1
440 GOSUB 770
500 PRINT
510 PRINT
520 FOR R = 1 TO 6
530   PRINT USING "THE TOTAL OF MONTH # IS $###,###",R,T(R)
540 NEXT R
550 FOR R = 1 TO 3
560   T = T + T(R)
570 NEXT R
580 PRINT
590    PRINT USING "THE TOTAL FOR THE FIRST THREE MONTHS IS
$###,###.##",T
600 FOR R = 4 TO 6
610   T1 = T1 + T(R)
620 NEXT R
630 PRINT
640   PRINT USING "THE TOTAL FOR THE LAST THREE MONTHS IS
$###,###.##",T1
650 T2 = T1 + T
660 PRINT
670 PRINT USING "THE TOTAL FOR THE 6 MONTHS IS
$###,###,###.##",T2
680 PRINT
690 FOR C = 1 TO 6
696   FOR R = 1 TO 6
697     T1(C) = T1(C) + A(R,C)
698   NEXT R
699   PRINT USING "THE TOTAL FOR DEPARTMENT NO. # IS
$###,###.##",C,T1(C)
700 NEXT C
701 GO TO 999
702 REM PRINTS ARRAY
703 PRINT:PRINT
704 PRINT USING "MONTH                          DEPARTMENT"
705 PRINT
706 FOR H = B1 TO D STEP S1
707 PRINT USING "          #  ",H;
708 NEXT H
709 PRINT
710 FOR R = B TO M STEP S
714 PRINT
715 PRINT USING F1$,R;
720   FOR C = B1 TO D STEP S1
730     PRINT USING F$, A(R,C);
740   NEXT C
750 NEXT R
760 RETURN
770 REM  COMPUTES TOTALS
775 FOR R = 1 TO 6
```

```
780   FOR C = 1 TO 6
785     T(R) = T(R) + A(R,C)
790   NEXT C
794 NEXT R
800 RETURN
900 DATA 20000,4000,900,40000,10000,6000
910 DATA 15000,2000,200,20000,8000,4000
920 DATA 30000,6000,500,20000,7000,3000
930 DATA 40000,6000,800,40000,10000,4000
940 DATA 20000,5000,600,30000,6000,5000
950 DATA 15000,1000,700,15000,10000,6000
999 END
```

8.

```
001 REM  THIS PROGRAMS USES SINGLE- AND DOUBLE-DIMENSIONAL
002 REM  ARRAYS TO CALCULATE AND PRINT THE FOLLOWING:
003 REM
004 REM   1.  TOTAL DOLLAR SALES FOR EACH SALES
005 REM       REPRESENTATIVE
006 REM   2.  TOTAL DOLLAR SALES FOR ALL SALES
007 REM       REPRESENTATIVES
008 REM   3.  TOTAL NUMBER OF EACH MODEL SOLD
009 REM   4.  TOTAL NUMBER OF ALL CARS SOLD
010 REM   5.  PERCENT OF TOTAL CARS SOLD BY EACH MODEL
011 REM
012 REM   M(5)   = ARRAY FOR PRICES OF CARS
020 REM   S$(5)  = ARRAY FOR SALES REPRESENTATIVES
030 REM   S(5,4) = ARRAY FOR NUMBER OF EACH MODEL SOLD BY
035 REM            EACH SALES REPRESENTATIVE
040 REM   T(5)   = ARRAY FOR TOTAL SALES FOR EACH
043 REM            REPRESENTATIVE
045 REM   N(5)   = ARRAY FOR TOTAL NUMBER OF EACH MODEL SOLD
050 REM   P(5)   = ARRAY FOR PERCENT OF TOTAL CARS SOLD BY
060 REM            EACH MODEL
070 REM    I, J, R, C  = LOOP CONTROL VARIABLES
080 REM
100 DIM M(5),S$(5),S(5,4),T(5),N(5),P(5)
105 REM STATEMENTS 110 - 198 LOAD THE ARRAYS
110 FOR I = 1 TO 4
120   READ M(I)
130 NEXT I
140 FOR J = 1 TO 5
150   READ S$(J)
160 NEXT J
170 FOR R = 1 TO 5
180   FOR C = 1 TO 4
190     READ S(R,C)
192   NEXT C
198 NEXT R
200 REM TOTALS $ SALES FOR EACH SALES REPRESENTATIVE
210 FOR R = 1 TO 5
230   FOR C = 1 TO 4
240   T(R) = T(R) + S(R,C)* M(C)
250   NEXT C
260 NEXT R
```

```
265 PRINT
270 FOR R = 1 TO 5
280  PRINT USING "\             \ SOLD
$###,###.##",S$(R),T(R)
290 NEXT R
300 REM TOTALS FOR ALL SALES REPRESENTATIVES
310 FOR R = 1 TO 5
320   T1 = T1 + T(R)
330 NEXT R
335 PRINT
340 PRINT USING "TOTAL OF ALL CARS SOLD =
$###,###,###.##",T1
400 REM  TOTAL NUMBER OF EACH MODEL SOLD
410 FOR C = 1 TO 4
420   FOR R = 1 TO 5
430     N(C) = N(C) + S(R,C)
440   NEXT R
450 NEXT C
455 PRINT
460 FOR J = 1 TO 4
470  PRINT USING " ##,### CARS OF MODEL # SOLD",N(J),J
480 NEXT J
500 REM TOTAL NUMBER OF ALL CARS SOLD
510 FOR J = 1 TO 4
520  T2 = T2 + N(J)
530 NEXT J
535 PRINT
540 PRINT USING "TOTAL NUMBER OF ALL CARS SOLD IS ##,###", T2
600 REM  COMPUTES PERCENT OF TOTAL CARS SOLD BY MODELS
610 FOR I = 1 TO 4
620  P(I) = N(I)/T2*100
630 NEXT I
635 PRINT
640 FOR I = 1 TO 4
650   PRINT USING "MODEL ## HAS ##% OF THE TOTAL CARS
SOLD",I,P(I)
660 NEXT I
800 DATA 8000,12000,15000,18000
810 DATA "JONES","THOMAS","CARLTON","BATES","HUGHES"
820 DATA 2,6,4,8,4,8,8,6,8,9,6,4,3,6,2,3,5,2,1,1
999 END
```

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

NAME____________________________

TEST

1. Match the terms on the right with the correct definitions.

_____a. Table with values in columnar locations only

_____b. Variables that store related numeric and string values that can be referenced with one variable name having single- or double-subscripts

_____c. The second subscript in an array, usually represented as moving down a page (vertical)

_____d. The first subscript in an array, usually represented as moving across a page (horizontal)

_____e. The number or variable that shows the location of one element in an array (enclosed in parentheses right after the array name)

_____f. To declare the name and size of an array to the computer

_____g. Table with values in locations of rows and columns

_____h. A list of related numeric or string values that are of the same type and use the same variable name

_____i. BASIC statement that is used to reserve space in the computer's memory for an array

_____j. A single position in an array - each is equivalent to a variable

1. Array
2. Column
3. DIM statement
4. Dimension
5. Double-dimension array
6. Element
7. Row
8. Single-dimension array
9. Subscript
10. Subscripted variables

2. Select true statements concerning subscripted variables and single-dimensional arrays by placing an "X" in the appropriate blanks.

_____a. An element in an array must be referenced as a constant

_____b. The subscript refers to the location of the element in storage

_____c. It is invalid to use the same variable as a subscripted variable and as an unscripted variable

_____d. Arrays provide a fast and orderly way of storing and retrieving large quantities of data

_____e. A(5) is referred to as A sub-5

_____f. Each element in an array can be used for any purpose a single variable can be used

_____g. String arrays are not allowed in most forms of BASIC

_____h. A(80) always has the value of 80

_____i. A list or table is referred to as an array

_____j. A subscript does not have to be enclosed in parentheses

3. Select true statements concerning the DIM statement by placing an "X" in the appropriate blanks.

_____a. The same variable can be dimensioned as many times as needed in a program

_____b. In the DIM statement, the maximum size of the array can be stated as a constant, variable, or arithmetic expression on most computers

_____c. The DIM statement should be for more elements than are actually needed in the program

_____d. The DIM statement is referred to as a dimension statement

_____e. An error message will be printed if the DIM statement is smaller than the number of elements used in the array in the program _____f. A DIM statement would be required for an array of 8 elements

_____g. It is a good programming practice to DIM all arrays

_____h. The DIM statement must be placed at the beginning of a program

_____i. The following statement is a valid DIM statement:

100 DIM A(80),B$(5),C(57)

_____j. The DIM statement is used to reserve space in the memory of the computer for arrays

4. Write FOR/NEXT loops to load an array, manipulate data, and print an entire array.

Write a separate FOR/NEXT loop for each of the following:

a. To load the following values into array X

4,7,3,5

____

____

____

____

____

____

b. To load the following values into array N$

Jones, Smith, Taylor

____

____

____

____

____

____

c. Add the values in array X together and assign total to variable T

____

____

____

____

____

____

d. Print array X, array N$, and T

____

____

____

____

____

____

5. Select true statements concerning double-dimensional arrays by placing an "X" in the appropriate blanks.

_____a. Two subscripts are required for a double-dimensional array

_____b. The following will manipulate data in the double-dimensional array in a horizontal manner:

```
100 DIM Z(5,6)
110 FOR R = 1 TO 5
120   FOR C = 1 TO 6
130     PRINT Z(R,C)
140   NEXT C
150 NEXT R
```

_____c. The first subscript of a double-dimensional array refers to the column

_____d. A nested loop is necessary to load, manipulate, and print data in a double-dimensional array _____e. The following will manipulate data in the double-dimensional array in a vertical manner:

```
100 DIM Z(5,6)
110 FOR C = 1 TO 6
120    FOR R = 1 TO 5
130       PRINT Z(R,C)
140    NEXT R
150 NEXT C
```

_____f. A DIM statement is required for all double-dimensional arrays

_____g. The same variable name can be used as a single- and double-dimensional array in the same program

_____h. The same variable name can be used as an unscripted variable and as a double-subscripted variable in the same program

_____i. The row and column maximum size must be specific positive integer numbers

_____j. The second subscript of a double-dimensional array refers to the row

6. Identify elements in single- and double-dimensional arrays by location and values.

7. Write the results of programs containing single- and double-dimensional arrays.

8. Write statements to dimension, load, manipulate, and process single- and double-dimensional arrays.

9. Flowchart, code, enter, and RUN programs containing single- and double-dimensional arrays.

(NOTE: If objectives 6 - 9 have not been accomplished prior to the test, ask your instructor when they should be completed.)

SUBSCRIPTED VARIABLES, THE DIMENSION STATEMENT, AND SINGLE- AND DOUBLE-DIMENSIONAL ARRAYS
UNIT X

ANSWERS TO TEST

1. a. 8
   b. 10
   c. 2
   d. 7
   e. 9
   f. 4
   g. 5
   h. 1
   i. 3
   j. 6

2. b.
   d.
   e.
   f.
   i.

3. c.
   d.
   e.
   g.
   i.
   j.

4. a.

```
100 DIM X(4)
110 FOR I = 1 TO 4
120   READ X(I)
130 NEXT I
800 DATA 4,7,3,5
999 END
```

b.

```
101 DIM N$(3)
210 FOR I = 1 TO 3
220   READ N$(I)
230 NEXT I
801 DATA "JONES","SMITH","TAYLOR"
999 END
```

c.

```
300 FOR I = 1 TO 4
310   T = T + X(I)
320 NEXT I
999 END
```

d.
```
400 FOR I = 1 TO 4
410   PRINT X(I)
420 NEXT I
430 FOR I = 1 TO 3
440   PRINT N$(I)
450 NEXT I
460 PRINT T
999 END
```

5. a.
   b.
   d.
   e.
   f.
   h.
   i.

6. Evaluated to the satisfaction of the instructor.

7. Evaluated to the satisfaction of the instructor.

8. Evaluated to the satisfaction of the instructor.

9. Evaluated to the satisfaction of the instructor.